“MODULACION Y DEMODULACION TIPO PCM, DELTA, ASK, FSK Y PSK COMO EQUIPOS DE LABORATORIO PARA LA ASIGNATURA SISTEMAS DE COMUNICACIONES 1”. Y MODULADOR EN CODIGO MANCHESTER COMO EQUIPOS DE LABORATORIO PARA LA ASIGNATURA SISTEMAS DE COMUNICACIONES 1”

CARLOS ANDRES ALVAREZ GOMEZ

ALEXANDER CUELLAR OSPINA

JOSE ANTONIO ARISTIZABAL MOTTA

CORPORACION UNIVERSITARIA AUTONOMA DE OCCIDENTE

DIVISION DE INGENIERIAS

PROGRAMA DE INGENIERIA ELECTRONICA

SANTIAGO DE CALI

2001

“MODULACION Y DEMODULACION TIPO PCM, DELTA, ASK, FSK Y PSK COMO EQUIPOS DE LABORATORIO PARA LA ASIGNATURA SISTEMAS DE COMUNICACIONES 1”. Y MODULADOR EN CODIGO MANCHESTER COMO EQUIPOS DE LABORATORIO PARA LA ASIGNATURA SISTEMAS DE COMUNICACIONES 1”

CARLOS ANDRES ALVAREZ GOMEZ

ALEXANDER CUELLAR OSPINA

JOSE ANTONIO ARISTIZABAL MOTTA

Trabajo para optar al titulo de
Ingeniero Electrónico

Director
IVAN HERRERA
Ingeniero Electrónico

CORPORACION UNIVERSITARIA AUTONOMA DE OCCIDENTE

DIVISION DE INGENIERIAS

PROGRAMA DE INGENIERIA ELECTRONICA

SANTIAGO DE CALI

2001

Nota de aceptación

Este trabajo fue presentado ante el comité de grado como requisito exigido por la Corporación Universitaria Autónoma de Occidente para optar al titulo de Ingeniero Electrónico.

Ing. Edwin Lopez

Jurado

Ing. Hernando Florez

Jurado

Santiago de Cali, 10 de Agosto de 2001

DEDICATORIA

A mis padres, Fernando y Floralba a quienes debo lo que soy. A ustedes dedico este logro, gracias por su apoyo.

A mis hermanos Paulo y David, muchas gracias por el apoyo que siempre me han brindado.

A Diana, quien me ha brindado su amor y colaboración en este trayecto de mi vida.

ANDRES

CONTENIDO

pág.

LISTA DE FIGURAS

pág.

LISTA DE TABLAS

pág.

LISTA DE ANEXOS

GLOSARIO

APLICACION: (Programa) Grupo de instrucciones que el Computador utiliza para realizar determinadas tareas.

CLAREAR: Poner a cero, es decir, cambiar a un estado de cero lógico.

LOOP: Lazo cerrado (bucle).

SETEAR: Poner a uno, es decir, cambiar a un estado de uno lógico.

POLLING: Observar una posición binaria para determinar si se presenta algún cambio de estado lógico.

PROTOCOLO: Conjunto de normas que los Computadores utilizan para comunicarse entre sí a través de una línea de red.

RESUMEN

Este proyecto surge de la necesidad de observar de manera practica las diferentes señales asociadas a cada uno de los métodos de codificación y modulación que se estudian durante la asignatura *“Sistemas de Comunicaciones 1”*. Con este propósito se construyen una serie de módulos que cumplen con este objetivo.

El componente principal de cada uno de los módulos es un microcontrolador, este se encarga de realizar la modulación y la demodulación de una señal de entrada utilizando distintos métodos de modulación o codificación. El usar un microcontrolador reduce la complejidad de los módulos, lo que los hace más sencillos de utilizar y de comprender.

Se crea una herramienta para fortalecer el laboratorio de Ingeniería Electrónica que a su vez es un sistema didáctico que facilita el aprendizaje de una forma concisa y clara para el estudiante.

INTRODUCCION

Cuando se inicia estudios relacionados con comunicaciones, más concretamente en la asignatura Sistemas de Comunicaciones 1, se observa a nivel general cierto inconformismo por parte de los estudiantes acerca de todas las tecnologías en los tipos de modulación tanto análoga como digital, debido a que estas solo se ven en el aspecto teórico más no practico, por esto, no se logra un claro reconocimiento de estas tecnologías de manera especifica.

Debido al gran avance tecnológico en el área de las comunicaciones, la universidad se ve en la obligación de cumplir con unos requisitos mínimos para su correcto aprendizaje. Es así, como los módulos propuestos en este trabajo se realizan con el objetivo de brindar una ayuda primaria a los estudiantes que cursan la asignatura Sistemas de Comunicaciones 1.

Así, se logran complementar los conocimientos teóricos en cuanto a codificación Manchester y Modulación FSK, PSK, ASK, PCM y Delta.
Se tomo entonces, la decisión de utilizar dos de las posibles combinaciones para técnicas de codificación y modulación:

- *Datos digitales, señales digitales*: en términos generales, el equipo para la codificación digital usando señales digitales es menos complicado y menos costoso que el equipo necesario para transmitir datos digitales modulando señales analógicas.

- *Datos analógicos, señales digitales*: la conversión de los datos analógicos en digitales permite la utilización de las técnicas más recientes de equipos de comunicación para transmisión digital, la cual tiene mucha mayor precisión que la transmisión analógica.

Se debe tomar en consideración que para efecto de manejo de la combinación de datos digitales y señales digitales se utilizan las técnicas de modulación digital, lo cual implica la modificación de uno o más de los tres parámetros fundamentales de una señal que se declara portadora: la amplitud, la frecuencia o la fase.

1. PLANTEAMIENTO DEL PROBLEMA

La universidad no posee ningún dispositivo para el desarrollo de laboratorios de comunicaciones, se hace necesario entonces adquirir una cantidad mínima de elementos para cumplir con ese objetivo, dichos elementos se encuentran pero no se ajustan específicamente a las necesidades del estudiante, debido a que incorporan muchas aplicaciones que son innecesarias y muy complejas en el momento del aprendizaje primario de las comunicaciones, además de hacer su costo mucho mayor.

2. JUSTIFICACIÓN

Debido a que la universidad carece de equipos para realizar prácticas de laboratorios enfocadas a observar el comportamiento de la codificación digital en comunicaciones, se busca implementar una serie de módulos que permitan observar técnicas como: la codificación Manchester, la modulación por desplazamiento de amplitud (ASK), la modulación por desplazamiento de frecuencia (FSK), la modulación por desplazamiento de fase (PSK), la modulación por codificación de pulsos (PCM) y la modulación Delta. Por ser estas técnicas las más usadas y las más importantes en comunicaciones digitales se hace necesario que los conocimientos adquiridos en este campo no sean solo teóricos sino que se pueda obtener una aplicación práctica.

El objetivo es realizar los módulos con la menor complejidad posible, teniendo en cuenta que los equipos que se encuentran en el mercado son demasiado costosos y lo que se busca es obtener módulos de bajo costo.

3. OBJETIVOS

3.1 OBJETIVO GENERAL

Desarrollo e implementación de un conjunto de módulos para laboratorio de la asignatura “Sistemas de Comunicaciones 1”.

3.2 OBJETIVOS ESPECIFICOS

1. Estudio previo de los diferentes sistemas de codificación de línea.

2. Estudio de la teoría de modulación y demodulación en señales digitales y análogas.

3. Evaluación de cada uno de los componentes que se usarán para el desarrollo de cada módulo.

4. Diseñar, implementar y probar cada uno de los módulos.

5. Redactar un manual de uso para cada módulo.

6. Desarrollar un ambiente gráfico para el manejo de los módulos que realizan modulación de señales digitales, en Visual Basic 6.0.

4. ANTECEDENTES

En la Corporación Universitaria Autónoma de Occidente, el laboratorio de electrónica no posee ninguno de los módulos a realizar; estos módulos no se comercializan a nivel nacional, se consiguen en el mercado externo, a unos precios altísimos y con aplicaciones bastante complejas; las cuales, no se adecuan para un aprendizaje primario en la asignatura Sistemas de Comunicaciones que es lo que se busca con este proyecto.

5. MARCO TEORICO

Para el manejo de los módulos a realizar se examinará el planteamiento teórico de cada una de las técnicas de codificación y modulación.

5.1 CODIGO MANCHESTER

Una técnica para modulación de datos digitales es el código Manchester.

En el código Manchester siempre hay una transición en la mitad del intervalo de duración de un bit, esta transición es la sincronización entre la emisión y recepción cuando se realiza una transmisión de datos, en este tipo de modulación la transición de un nivel bajo de voltaje a un nivel alto de voltaje representa un 1 lógico, y una transición de un nivel alto a un nivel bajo de voltaje representara un 0 lógico.

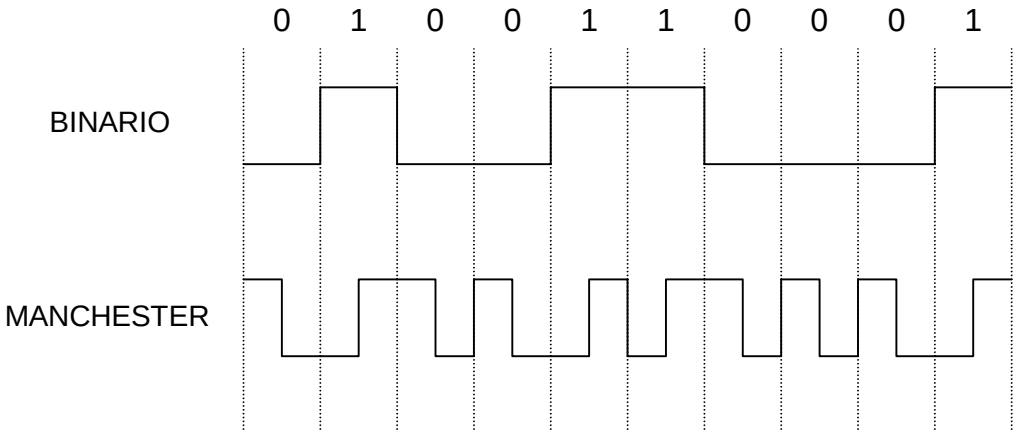


Figura 1 *Formas de onda Binaria y Manchester*

Las ventajas de esta técnica son varias, entre ellas esta la sincronización, debido a la transición que ocurre durante el intervalo de duración correspondiente a un bit, el receptor puede sincronizarse usando dicha transición. Además en la detección de errores, estos se pueden determinar con una ausencia de la transición esperada en la mitad del intervalo. Este código se utiliza en la especificación de la normalización IEEE 802.3 para transmisión en redes LAN con bus CSMA / CD.

5.2 MODULACION FSK

FSK, (*Frequency Shift Keying*), modulación por desplazamiento de frecuencia; este tipo de modulación es similar a la modulación en frecuencia (FM) convencional. La diferencia radica en que la señal modulante es un flujo de pulsos binarios que varía entre dos niveles de voltaje.

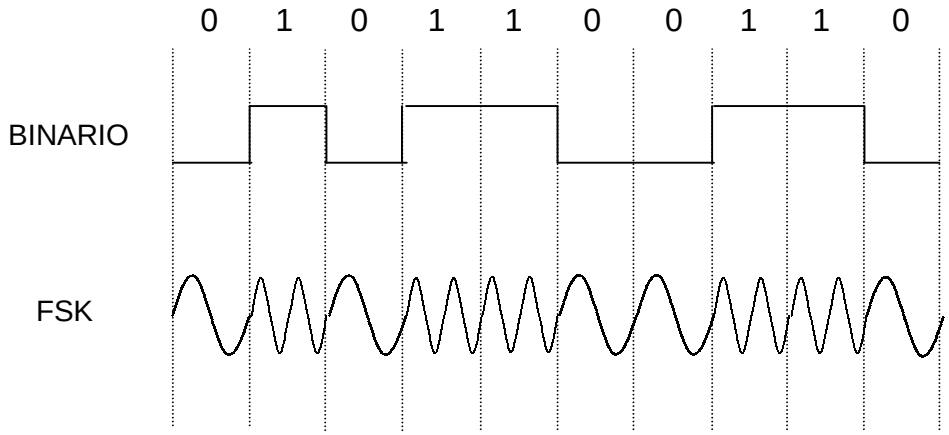


Figura 2 *Formas de onda Binaria y FSK*

Para la transmisión a partir de una entrada binaria, la salida del modulador FSK cambia de acuerdo a la señal de entrada binaria que puede ser un uno lógico o cero lógico o viceversa. La señal transmitida por el modulador FSK es una señal análoga que se desplaza únicamente entre dos frecuencias: una frecuencia de marca que corresponde a un uno lógico y una frecuencia de espacio que corresponde a un cero lógico.

Esto significa que hay un cambio en la salida cada vez que la condición lógica de entrada cambia, esto suele hacerse con un VCO (oscilador controlado por voltaje). En la recepción de la señal FSK es común utilizar un circuito de fase cerrada (PLL), conforme cambia la entrada del PLL entre las frecuencias de marca y espacio, el voltaje de salida del comparador de fase sigue el cambio de frecuencia; Y como solo hay dos frecuencias de entrada también hay solo dos voltajes de salida, uno representa un uno lógico y el otro representa un cero lógico.

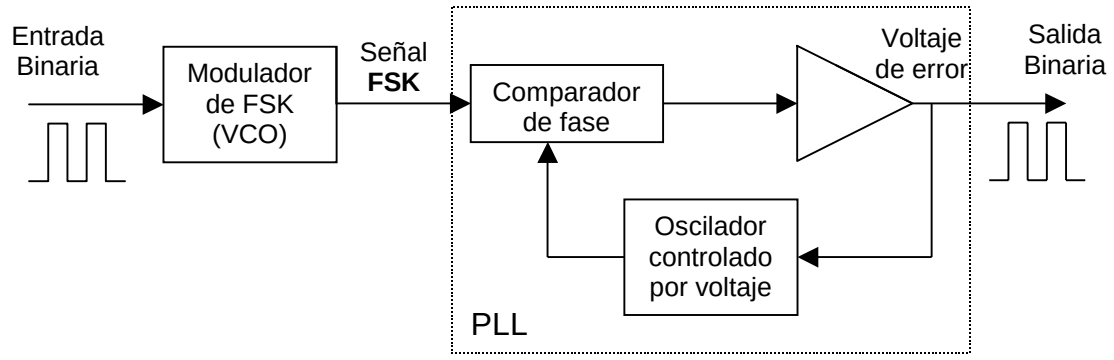


Figura 3 *Diagrama de bloques de un transmisor y un receptor FSK típicos*

5.3 MODULACION PSK

PSK, (*Phase Shift Keying*), modulación por desplazamiento de fase; con el PSK dos fases de salida son posibles para una sola frecuencia de la portadora. Una fase de salida representa un uno lógico y la otra representa un cero lógico. Con el cambio de la señal de entrada digital la fase de la portadora de salida cambia entre dos ángulos que están desfasados 180º.

En la transmisión el modulador actúa como un conmutador para invertir la fase. Dependiendo de la condición lógica de la entrada digital la portadora es transferida a la salida, ya sea en fase o 180º fuera de fase con respecto a sí misma, luego la señal pasa por un filtro pasa-banda para generar la salida PSK.

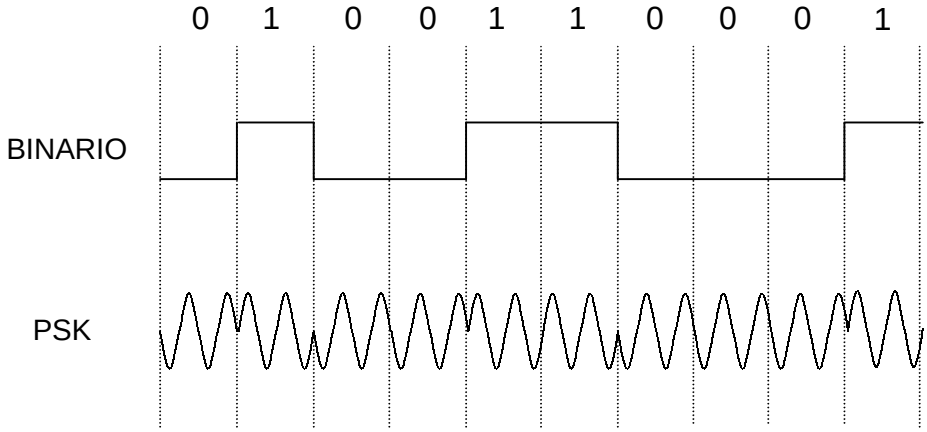


Figura 4 *Formas de onda Binaria y PSK*

En la recepción la señal PSK de entrada llega a dos circuitos, el circuito de recuperación de la portadora cuya función es detectar y regenerar una señal que estará sincronizada con la portadora de la entrada, y al circuito de

modulación balanceada el cual se encarga de generar una señal que es el producto de las dos entradas (la señal PSK y la señal que sale del circuito de recuperación de portadora), por último la señal pasa por un filtro pasa banda que genera los datos binarios de la señal original.

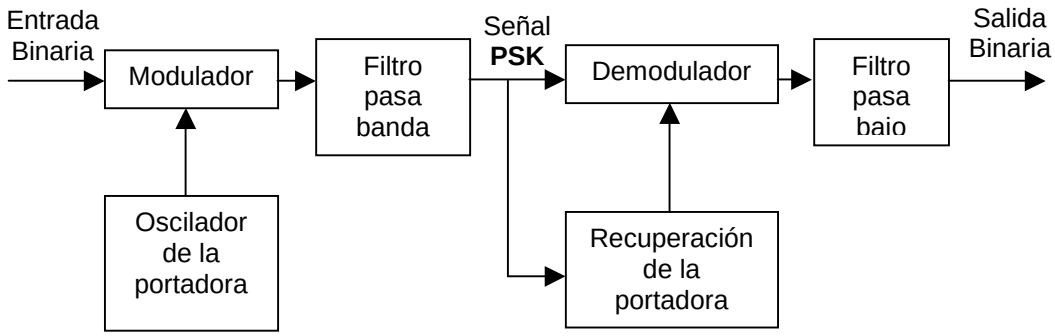


Figura 5 *Diagrama de bloques de un transmisor y un receptor PSK típicos*

5.4 MODULACION ASK

ASK, (*Amplitude Shift Keying*); transmisión por desplazamiento de amplitud, también se conoce como OOK, (*On Off Keying*); transmisión por cierre y

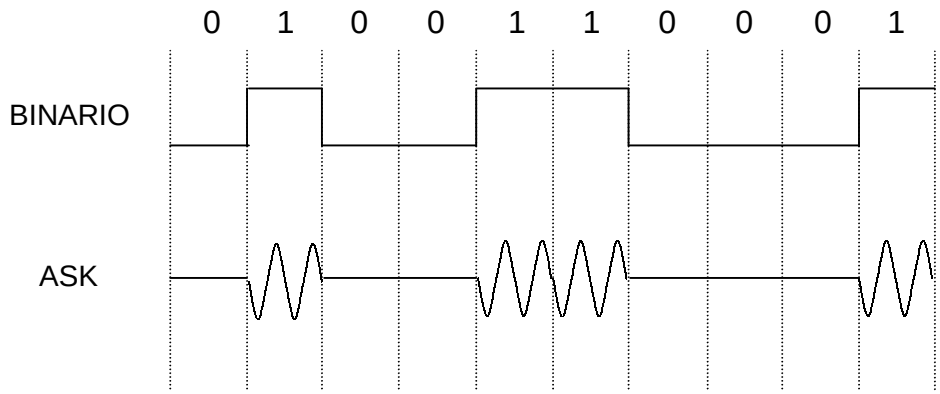


Figura 6 *Formas de onda Binaria y ASK*

apertura, la cual consiste en activar o desactivar (conmutar) una portadora sinusoidal con una señal binaria unipolar.

Se usa como fuente una señal binaria para ser transportada en una portadora cuya amplitud variará entre dos niveles. Los dos valores binarios se representan mediante dos amplitudes diferentes de la portadora. La amplitud se desplaza entre un valor máximo y un valor mínimo los cuales son un uno lógico y un cero lógico respectivamente. Uno de los dígitos binarios se representa mediante la presencia de la portadora a amplitud constante, y el otro mediante la ausencia de portadora. La transmisión de radio del código Morse es un ejemplo de esta técnica.

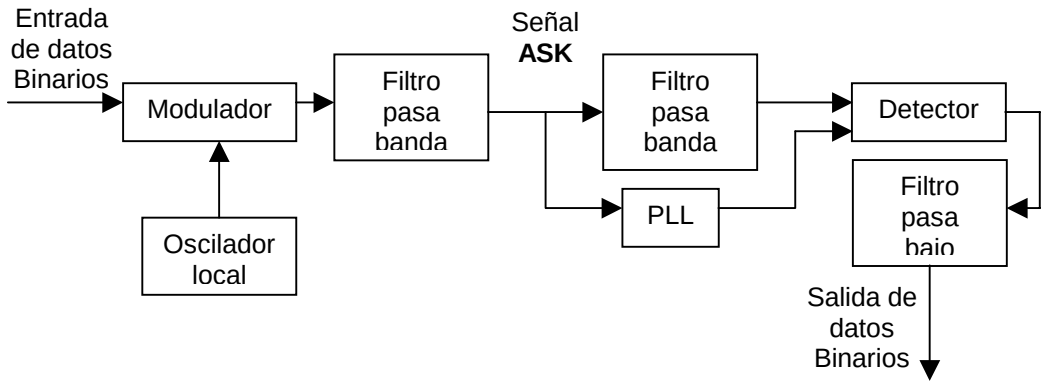


Figura 7 *Diagrama de bloques de un transmisor y un receptor ASK típicos*

5.5 MODULACION PCM

PCM, (*Pulse Code Modulation*), modulación por codificación de pulso; esta técnica es esencialmente una conversión de señal análoga a digital, donde la información contenida en la muestra instantánea de una señal análoga es

representada por una palabra digital, donde cada una corresponde a cierto nivel de amplitud de la señal análoga. Con este tipo de modulación se toma la señal análoga de entrada, se le asigna un ancho de banda a la señal que se va a codificar, esto se realiza utilizando un filtro pasa banda, luego la señal filtrada pasa por un circuito de muestreo y retención que periódicamente toma una muestra de la señal convirtiéndola en una señal PAM multinivel, modulación por amplitud de pulsos, este es un término de ingeniería usado para describir la conversión de una señal análoga a una de tipo pulso donde la amplitud del pulso denota la información análoga, después esta señal entra a un conversor análogo-digital cuya particularidad es transformar las muestras PAM a datos binarios para luego ser transmitidos serialmente, el medio de transmisión puede ser por ejemplo cable metálico o fibra óptica.

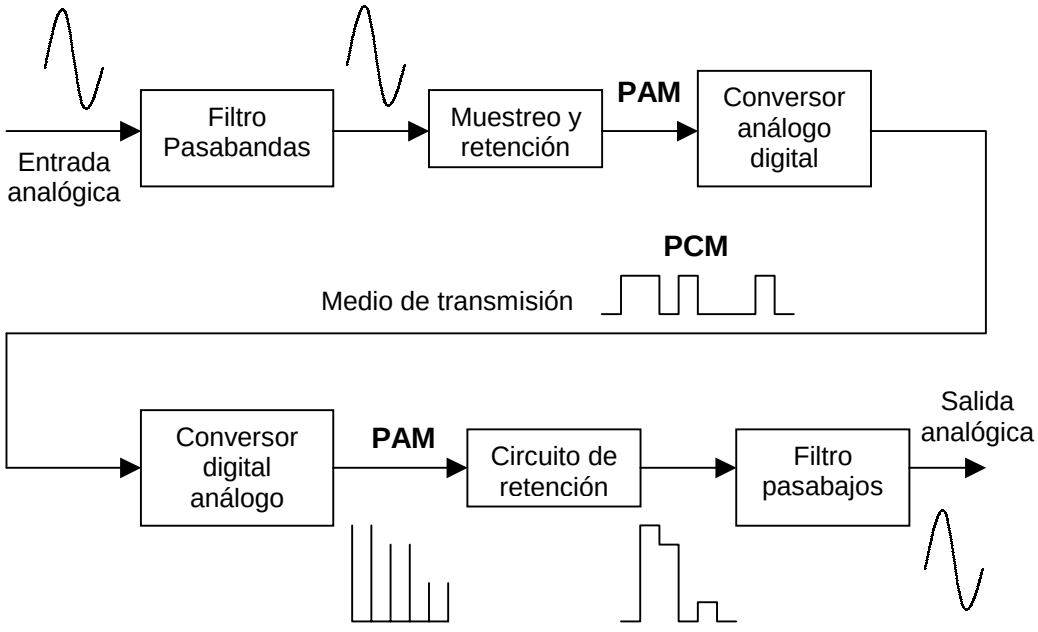


Figura 8 *Diagrama de bloques de un transmisor y un receptor PCM típicos*

En la recepción se obtiene la información realizando el proceso contrario, es decir, primero la señal binaria que fue transmitida pasa por un conversor digital-

análogo que la transforma a una señal PAM multinivel, luego el circuito de retención y un filtro pasa bajo convierte la señal PAM a su forma análoga original. PCM es muy popular debido a las muchas ventajas que ofrece entre las cuales están; elementos digitales relativamente baratos para su uso, las señales PCM derivadas de todos los tipos de origen análogo (audio, vídeo, etcétera) pueden ser mezcladas con señales de datos y transmitidas sobre un sistema de comunicación digital común de alta velocidad.

5.6 MODULACION DELTA

Esta técnica de modulación utiliza como principio un código PCM de bit sencillo para lograr la transmisión, lo que significa que para la transmisión DELTA se transmite solo un bit, que solamente indica si la muestra es mayor o menor que la anterior. Para esto existe un algoritmo sencillo, si la muestra actual es menor que la anterior se transmite un cero lógico de lo contrario se transmite un uno lógico.

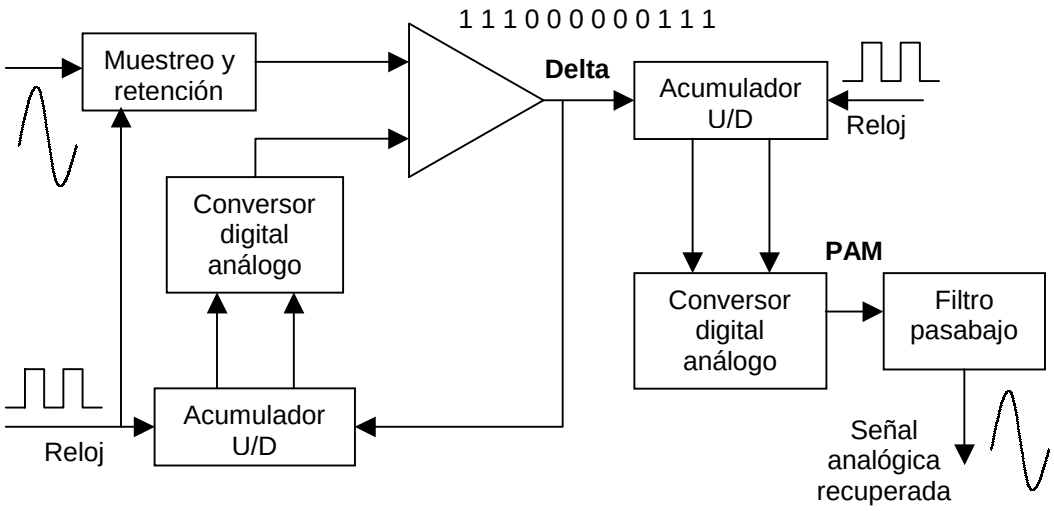


Figura 9 *Diagrama de bloques de un transmisor y un receptor DELTA típicos*

Para la transmisión, en la entrada análoga se hace un muestreo y se convierte a una señal PAM, la cual se compara con la salida de un conversor digital-análogo, cuya magnitud corresponde al bit de la muestra anterior que se ha almacenado por medio de un acumulador binario que se incrementa o decrementa dependiendo de si la muestra anterior fue mayor o menor que la muestra actual, este acumulador se sincroniza con el muestreo de la entrada, por lo tanto el acumulador se actualizará después de cada comparación.

Para la recepción existe un acumulador ascendente-descendente que recibe la señal, incrementándose o decrementandose de acuerdo al uno lógico o cero lógico recibido. La señal que genera el acumulador maneja un conversor digital-análogo que genera nuevamente la señal PAM pasándola por un filtro pasa bajo para recuperar la señal original enviada.

6. METODOLOGÍA

1. Revisión bibliográfica sobre los aspectos teóricos y técnicos propios de cada uno de los módulos.

2. Profundización en el manejo de los microcontroladores PIC de la empresa *Microchip Technology Inc*.

3. Selección de los componentes electrónicos adecuados para cada uno de los módulos a diseñar.

4. Diseño e implementación de cada uno de los módulos bajo supervisión del respectivo orientador.

5. Montaje final y realización de pruebas de desempeño de cada uno de los módulos.

6. Documentación del proyecto y elaboración de manuales de uso para cada uno de los módulos.

7. EL MICROCONTROLADOR PIC

7.1 DEFINICION

El Microcontrolador es un circuito integrado programable que contiene todos los elementos de una Computadora. Se utiliza para controlar y realizar una tarea especifica. Ya que en su memoria se graba solamente un programa destinado para una aplicación determinada; Sus líneas o pines de entrada-salida soportan la conexión de cualquier dispositivo digital que requiera para censar y controlar la aplicación.

7.2 ARQUITECTURA

El microcontrolador tiene todos los componentes de una Computadora, pero con características que no pueden alterarse.

Las partes de un microcontrolador son:

- Procesador
- Memoria no volátil para el programa
- Memoria de lectura-escritura para los datos
- Líneas de entrada-salida para los periféricos:

- Comunicación paralelo
- Comunicación serie
- Otros puertos de comunicación.
- Recursos auxiliares:
- Reloj
- Temporizador
- Perro Guardián (Watchdog)
- Conversor AD-DA
- Protección ante fallos de alimentación
- Estado de reposo o de bajo consumo

7.2.1 El Procesador. La necesidad de obtener mayor velocidad en la ejecución de instrucciones ha ocasionado la implementación de la arquitectura *Harvard* frente a la tradicional arquitectura *Von Newman*. Esta ultima se caracteriza por utilizar una memoria, donde se guardan datos e instrucciones a través de un solo bus de datos e instrucciones.

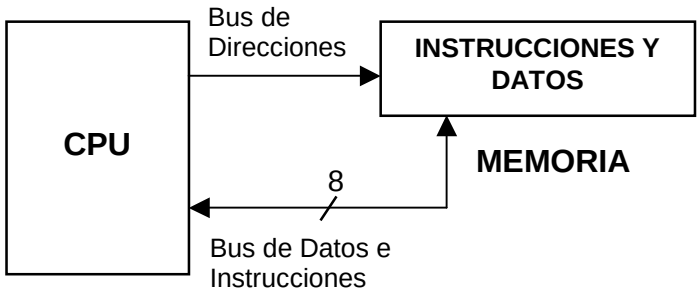


Figura 10 *Diagrama de bloques de la arquitectura Von Newman*

En la arquitectura Harvard el bus de datos es independiente del bus de instrucciones. Esto tiene como consecuencia la necesidad de una memoria

para las instrucciones y otra para los datos, y cada una con capacidades diferentes según su necesidad.

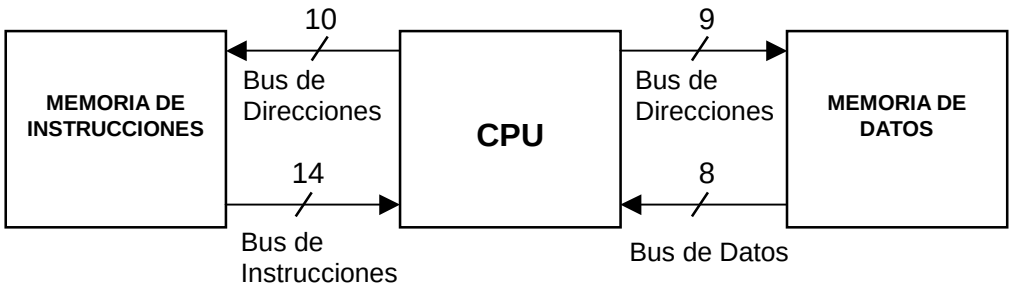


Figura 11 *Diagrama de bloques de la arquitectura Harvard*

El procesador de los microcontroladores PIC responde a la arquitectura RISC (*Reduced Instruction Set Computer*), que se identifica por poseer un set de instrucciones reducido y simple, así de esta manera la mayor parte de las instrucciones se ejecutan en un ciclo de maquina.

Otra técnica importante que aumenta el rendimiento del microcontrolador es el paralelismo, que consiste en la segmentación del procesador (*pipe-line*), descomponiéndolo en etapas para procesar una instrucción diferente en cada una de ellas y trabajar con varias a la vez.

7.2.2 Memoria de programa. En la memoria de programa se almacenan las instrucciones del programa que ejecutara el microcontrolador. Como el programa a ejecutar debe estar grabado de forma permanente, se utilizan cinco versiones de memoria diferentes. 1) La memoria ROM, es una memoria en la cual se graba el programa durante su proceso de fabricación. 2) La memoria

EPROM, en esta se graba el programa mediante diferentes niveles de voltaje y puede ser borrado utilizando luz ultravioleta. 3) La memoria OTP, (*one time programmable*), es programable solo una vez y no se puede borrar ni utilizar para otro programa. 4) La memoria EEPROM, es programable y borrable eléctricamente, donde este proceso puede realizarse tantas veces como sea necesario. 5) La memoria FLASH, es muy similar a la memoria EEPROM pero con menor consumo de energía y mayor capacidad.

7.2.3 Memoria de datos. Los datos varían continuamente, y esto exige que la memoria sea de lectura y escritura, por lo que la memoria RAM estática (SRAM) es la que más se adecua a los requerimientos, el único inconveniente es que sea una memoria volátil. Hay microcontroladores que tienen como memoria de datos el tipo EEPROM, para que así de esta manera un corte en el suministro de alimentación no ocasione la perdida de información.

7.2.4 Líneas de entrada / salida. A excepción de las dos líneas de alimentación y otras dos para el cristal de cuarzo que regula la frecuencia de trabajo, y una mas para el reset, las líneas restantes del microcontrolador soportan la comunicación con los periféricos externos que controla.

Las líneas de entrada-salida están agrupadas en conjuntos llamados *puertos,* por los que se recibe o transmite información de manera paralela. Existe

microcontroladores que trabajan de manera serial, o implementan protocolos de comunicación como el I^2C o el USB, etc.

7.2.5 Recursos auxiliares. Según las posibles aplicaciones a las que se orienta el fabricante del microcontrolador, incorpora una serie de complementos en cada modelo de microcontrolador supliendo la necesidad de cada cliente o empresa que necesite un microcontrolador. Los recursos más comunes son los siguientes:

- Circuito de reloj, que genera la sincronización para el funcionamiento del sistema.

- Temporizadores, que controlan tiempos y ratas de muestreo, etc.

- Perro Guardián (Watchdog), el cual provoca un reinicio cuando el programa se bloquea.

- Conversores AD y DA, con los que se reciben y envían señales de tipo análogo.

- Comparadores Analógicos, que verifican el valor de una señal análoga.

- Sistema de Protección Contra Fallos de Alimentación.

- Estado de Reposo, en el que el sistema con casi todos sus elementos apagados y el consumo de energía se reduce al mínimo.

7.3 RISC

Todos los microcontroladores PIC poseen la arquitectura RISC que significa Computador con set de instrucciones reducido. Lo que implica que el numero de instrucciones que el microcontrolador ejecutara es pequeño, Y posee las siguientes características:

- Las instrucciones son simples y rápidas: Debido a su poca complejidad, pueden ser ejecutadas en un solo ciclo de maquina, exceptuando las instrucciones de salto que tardan el doble.

- Las instrucciones son ortogonales: Solo tienen restricciones en el uso de operandos. Cualquier instrucción puede usar cualquier operando.

- El tamaño de las instrucciones y los datos es constante: Todas las instrucciones y los datos tienen 14 bits y 8 bits respectivamente. La arquitectura *Harvard* separa la memoria de instrucciones de la memoria de datos, pudiendo tener un tamaño diferente.

7.4 TIPOS DE FORMATO

Para los microcontroladores utilizados en este trabajo, dicho formato se divide en diferentes campos de bits, donde cada uno hace referencia a operandos o elementos que maneja la instrucción. A continuación se describen dichos campos:

- Campo del código OP (2 bits): Este campo define la operación que realiza la instrucción.

- Campo de los operandos fuente (f) y destino (d): Definen los registros que actúan como operandos en la instrucción, y referencian su ubicación en la memoria de datos.

- Campo de operando inmediato o literal (k): Como es obvio contiene el valor de un operando inmediato.

- Campo de referencia a un bit (b): Es un campo de tres bits que indica la posición de un bit concreto dentro de un registro de ocho bits.

- Campo de la dirección de salto: En las instrucciones CALL y GOTO existe un campo de bits que contiene la dirección de la siguiente instrucción que hay que ejecutar. Este campo se carga en el PC (Contador de programa) en las instrucciones de salto incondicional.

7.4.1 Operaciones orientadas a manejar registros de tamaño byte. El formato se divide en tres campos:

- Campo del código OP de 6 bits
- Campo de la dirección del operando fuente de siete bits
- Campo que define el operando destino de 1 bit

La sintaxis es: mnemónico f, d. Cuando d = 1 el registro destino coincide con el registro fuente.

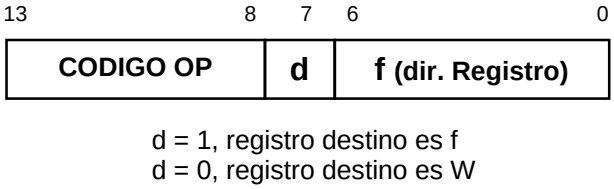


Figura 12 *Formato de instrucción orientada a manejar registros de tamaño byte*

7.4.2 Operaciones orientadas a manejar bits. Este formato se divide en los siguientes campos:

- Campo del código OP de 4 bits.
- Campo de dirección del registro fuente (7 bits).
- Campo de posición del bit en el registro (3 bits).

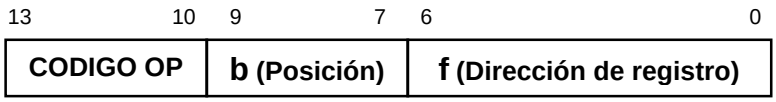


Figura 13 *Formato de la instrucción orientada a manejar bits*

7.4.3 Operaciones que manejan un valor inmediato o literal. En este formato se manejan solamente dos campos, ya que todos estos valores se cargan en el registro de trabajo W.

- Campo del código OP con 6 bits
- Campo del valor inmediato (k) con 8 bits

7.4.4 Operaciones incondicionales de control de flujo del programa. Este tipo de instrucciones actúan sobre el contador de programa, con lo que se rompe la secuencia del programa. Este formato se compone de solo dos campos:

- Campo del código OP de 3 bits
- Campo de la dirección de salto que se carga en el PC

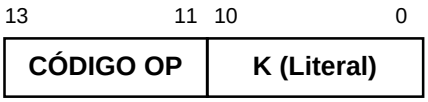


Figura 14 *Formato de la instrucción de salto incondicional*

7.4.5 Operaciones de salto condicional. Las instrucciones de salto para los PIC son pocas pero muy efectivas. Concretamente solo se salta una instrucción, la siguiente a la condición. Existen instrucciones de salto que manejan registros de un byte.

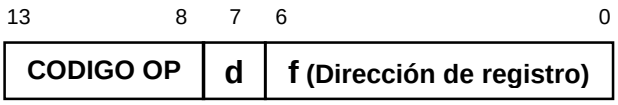


Figura 15 *Formato de la instrucción de salto condicional*

7.5 INSTRUCCIONES QUE MANEJAN REGISTROS

Estas instrucciones responden a la sintaxis "*mnemónico f, d*", siendo *f* y *d* los operandos fuente y destino implementados por registros de 8 bits en la memoria de datos. El registro *f* viene referenciado por una de dirección de 7 bits, mientras que el destino sólo por un bit, que al tener un valor cero dirige el resultado de la operación al registro de trabajo W, y al tener el valor uno dirige el resultado de la operación al registro fuente f.

Sintaxis	Operación	Ciclos	Señalizadores
ADDWF f,d	Suma W y f	1	C, DC, Z
ANDWF f,d	AND W con f	1	Z
CLRF f	Pone todos los bits de f a cero	1	Z
CLRW	Pone todos los bits de W a cero	1	Z
COMF f,d	Complemento de f	1	Z
DECF f,d	Decrementa f	1	Z
INCF f,d	Incrementa f	1	Z
IORWF f,d	OR entre W y f	1	Z
MOVF f,d	Mueve f	1	Z
MOVWF f	Mueve W y f	1	
NOP	No operación	1	
RLF f,d	Rota f a la izquierda a través del carry	1	C
RRF f,d	Rota f a la derecha a través del carry	1	C
SUBWF f,d	Resta W a f	1	C, DC, Z
SWAPF f;d	Intercambia nibbles	1	
XORWF f,d	XOR de W con f	1	Z

Tabla 1 *Instrucciones que manejan registros de ocho bits*

7.6 INSTRUCCIONES QUE MANEJAN BITS

Solo existen dos instrucciones en este grupo. Una de ellas pone a uno (bsf) en cualquier bit de un registro, mientras que la otra pone a cero (bcf).

Sintaxis	Operación	Ciclos	Señalizadores
BCF f,b	Borra un bit de f	1	
BSF f,b	Pone uno en el bit de f	1	

Tabla 2 *Instrucciones que manejan un bit determinado de un registro*

7.7 INSTRUCCIONES DE SALTO (SKIP)

Solo existen cuatro instrucciones de salto incondicional en los PIC. Dos de ellas revisan el bit de un registro determinado y según su valor (1 o 0), saltan o no. Las otras dos instrucciones incrementan o decrementan un registro y la posibilidad de salto se efectúa si el valor del registro es cero. Cuando no se cumple con la condición, no se efectúa el salto y la instrucción dura un ciclo de maquina. En caso de salto la instrucción dura dos ciclos de maquina.

Sintaxis	Operación	Ciclos	Señalizadores
BTFSC f,d	Explora un bit de f, y salta si vale cero	1(2)	
BTFSS f,d	Explora un bit de f, y salta si vale uno	1(2)	
DECFSZ f,d	Decremebta f y si es cero, salta	1(2)	

Tabla 3 *Instrucciones que manejan un salto condicional*

7.8 INSTRUCCIONES QUE MANEJAN OPERANDOS INMEDIATOS

Se trata de instrucciones que realizan operaciones con un valor inmediato de ocho bits que se proporciona dentro de la instrucción. Donde k es el valor inmediato.

Sintaxis	Operación	Ciclos	Señalizadores
ADDLW k	Suma inmediata con W	1	C, DC, Z
ANDLW k	AND inmediato con W	1	Z
IORLW k	OR inmediato con W	1	Z
MOVLW k	Mueve a W un valor inmediato	1	
SUBLW k	Resta W a un inmediato	1	C, DC, Z
XORLW k	OR exclusiva con W	1	

Tabla 4 *Instrucciones que manejan operandos inmediatos*

7.9 INSTRUCCIONES DE CONTROL Y ESPECIALES

En este grupo se incluyen instrucciones que rompen la secuencia normal de un programa porque alteran el contenido del PC (Contador de programa) y las instrucciones especiales.

La instrucción GOTO carga en el PC la dirección de la nueva instrucción.
La instrucción CALL llama a una subrutina, pero antes guarda la dirección actual del PC y luego carga la nueva dirección en el PC. De esta manera el retorno de la subrutina se hace cargando de nuevo la dirección de partida en el PC. Para realizar un retorno de una subrutina se pueden emplear dos instrucciones. La mas utilizada es RETURN, que se limita a extraer el valor que se cargara en el PC. Otra más compleja es RETLW k, que hace lo mismo que RETURN y además carga en W el valor inmediato k. Es decir, devuelve un parámetro desde la subrutina.

Para el final de las interrupciones hay una instrucción llamada RETFIE, la cual consiste en cargar el PC con el ultimo dato de la pila y pone el bit GIE = 1, pues este bit es el habilitador de la interrupción global del microcontrolador, pero se coloca en cero cuando entra a una interrupción para evitar que otras interrupciones se produzcan mientras es atendida la interrupción actual.

En cuanto a las instrucciones especiales, se incluyen dos: CLRWDT y SLEEP. La primera reinicia el contenido del perro guardián, ya que si se desborda provoca un reset. La instrucción SLEEP hace que el procesador entre a

trabajar en modo de reposo o de bajo consumo. Detiene el oscilador y el procesador queda inactivo.

Sintaxis	Operación	Ciclos	Señalizadores
CALL k	Llama a una subrutina k	2	TO#, PD#
CLRWDT	Borra o reinicia el perro guardian	1	
GOTO k	Salto incondicional	2	
RETFIE	Retorno de interrupción (GIE=1)	2	
RETLW k	Retorno de subrutina y carga W=k	2	
RETURN	Retorno de subrutina y carga W=k	2	
SLEEP	Modo de reposo	1	TO#, PD#

Tabla 5 *Intrucciones de control de flujo del programa y especiales*

8. DISEÑO E IMPLEMENTACION DE LOS MODULOS

A continuación se detallaran los diferentes aspectos que se tienen en cuenta durante el desarrollo e implementación de cada uno de los módulos.

8.1 DEFINICIONES

8.1.1 Codificación de línea. Los datos binarios se representan en varios formatos de señalización de bits serial llamados *códigos de línea*.

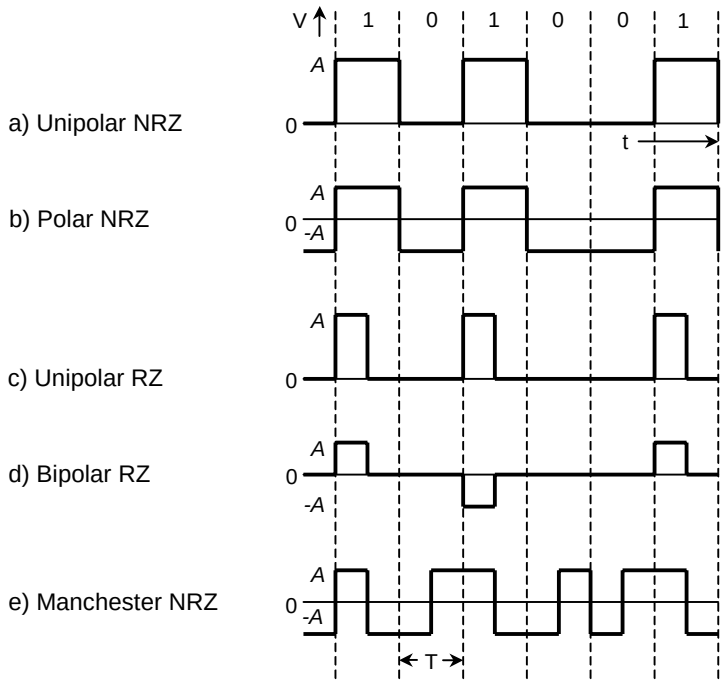


Figura 16 *Formatos de señalización binaria*

Algunos códigos de línea se muestran en la Figura 16. Existen dos categorías principales dentro de la codificación de línea: *con retorno a cero* (RZ) y *sin retorno a cero* (NRZ). Con la codificación RZ, la forma de onda retorna a un nivel de cero voltios, durante una fracción del intervalo de bit (usualmente la mitad). La forma de onda para la codificación de línea se puede clasificar aun más de acuerdo a los niveles de voltaje utilizados para representar el dato binario. Algunos ejemplos son:

- Señalización Unipolar: En el cual se representa un 1 binario con un nivel alto (+A voltios) y un 0 binario con un nivel de cero voltios.

- Señalización Polar: Se utiliza un voltaje positivo o negativo para representar un 1 binario o un 0 binario.

- Señalización Bipolar (seudoternaria): Se representa un 1 binario por medio de valores alternadamente positivos y negativos. El 0 binario es representado por un nivel de cero voltios.

- Señalización Manchester: Una transición a mitad de bit, de nivel negativo a positivo representa un 0 binario, y una transición a mitad de bit de nivel positivo a negativo representa un 1 binario.

8.1.2 Modulación. Modulación es el proceso de codificar la información original para transformarla en una señal modulada, es decir, se implanta la información original, (señal moduladora), en una señal portadora de frecuencia f_c, mediante la introducción de cambios de amplitud, frecuencia o fase en la

misma. La finalidad de la modulación es ayudar a la información a pasar a través del medio de transmisión (canal) de manera confiable.

El diagrama de bloques de la Figura 17 representa un sistema de comunicación, en el que se pueden observar tres subsistemas principales: *el transmisor, el canal* y *el receptor.*

En el transmisor un bloque procesador de señales recibe la información de entrada *m(t)* y según sean los datos este realiza cambios en la señal portadora *s(t)* (Modulación).

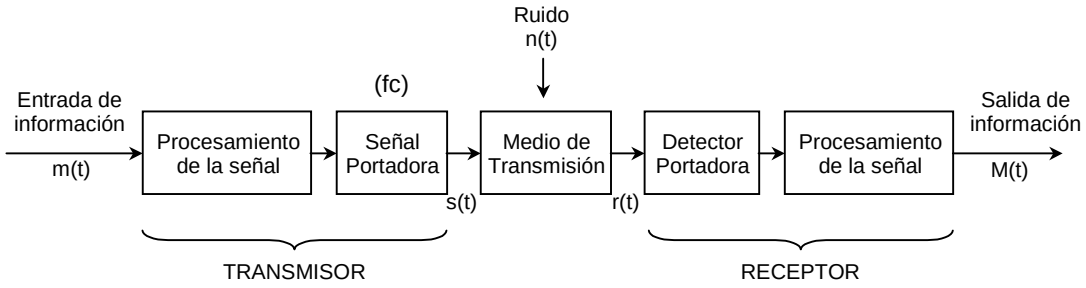


Figura 17 *Sistema de Comunicación*

La señal modulada es entonces enviada al canal (medio de transmisión), el cual se encarga de hacerla llegar hasta el receptor, es en esta parte que el ruido *n(t)* puede dañar la señal modulada. En el receptor la señal portadora es separada de la información, esta ultima es enviada al procesador de señales y este entrega una estimación de la información original *M(t).*

8.2 CARACTERISTICAS DE LOS MODULOS

Concretamente en el caso de los módulos diseñados para los sistemas ASK, FSK y PSK se puede decir que se utiliza codificación de línea *RZ polar*, ya que la señal portadora por ser un tren de pulsos retorna a un nivel cero varias veces durante el intervalo de un bit. En estos sistemas el manejo de los estados binarios se hace de la siguiente manera:

- FSK, se manejan dos frecuencias para representar un 0 binario o un 1 binario.
- PSK, se utilizan dos fases (0º y 180º) para representar un 0 binario o un 1 binario.
- ASK, un 0 binario se representa con un nivel bajo de voltaje y un 1 binario se representa con un tren de pulsos a una frecuencia determinada.

Otro de los módulos utiliza una codificación de línea diferente como es la señalización *Manchester NRZ*, el manejo que se da a cada uno de los estados binarios incluye una transición de estado a la mitad del periodo de un bit.

Los dos módulos PCM y DELTA utilizan codificación de línea del tipo *NRZ unipolar*, ya que estos son tipos de modulación análoga en los que no se hace manejo de una señal digital a través de una señal portadora.

8.3 COMPOSICION DE LOS MODULOS

Los diferentes módulos están compuestos por dos etapas:

1. Etapa de transmisión, la cual se encarga de realizar la modulación o codificación de la señal de entrada, y en la que están ubicados una serie de puntos de prueba, donde por medio de un osciloscopio se pueden observar las distintas formas de onda asociadas a cada tipo de modulación.
2. Etapa de recepción, esta es la encargada de la demodulación o decodificación de la señal enviada por el transmisor, esta etapa también posee una serie de puntos de prueba, donde se puede obtener la señal demodulada y otras señales diferentes como la de sincronización con el Osciloscopio.

Las señales de entrada dependen del tipo de módulo, de esta manera los módulos que realizan modulación de señales digitales utilizan como señal de

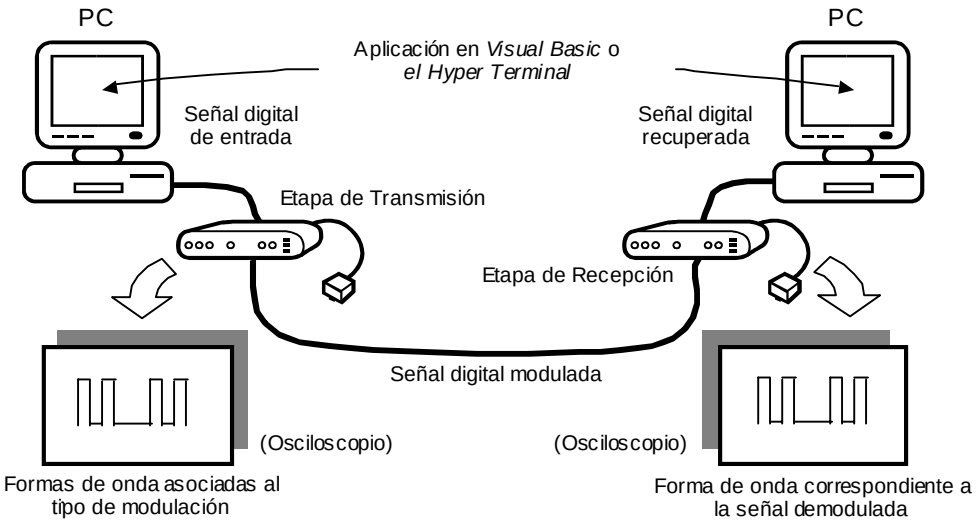


Figura 18 *Esquema general de los módulos que realizan modulación de señales digitales*

entrada, la enviada por un Computador, formada por una trama de datos serial en formato RS-232. De igual forma la señal digital recuperada es recibida en otro Computador, como se muestra en la Figura 18. Para visualizar los datos enviados o recuperados en los Computadores se utiliza una aplicación diseñada en VISUAL BASIC (AdCom) o con la misma finalidad también se puede utilizar el *Hyper-Terminal* de Windows.

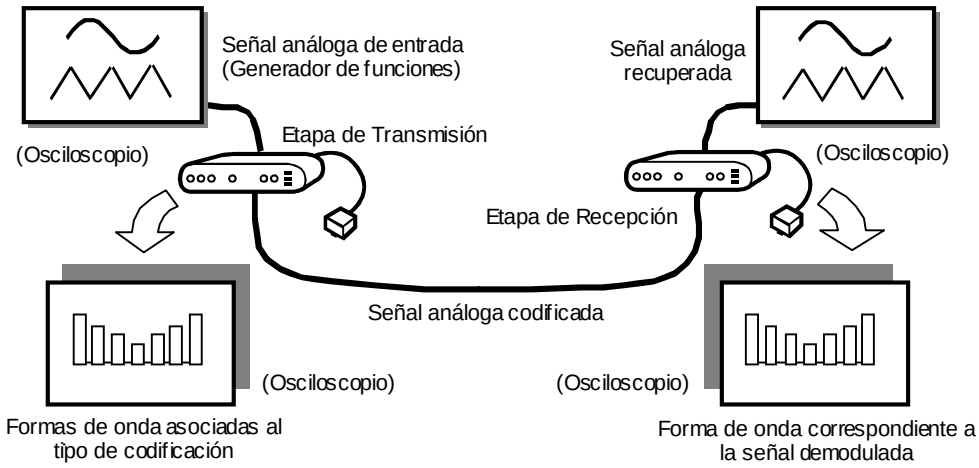


Figura 19 *Esquema general de los módulos de codificación de señales análogas*

En el caso de los módulos de codificación de señales análogas se utiliza como entrada un generador de funciones, del cual se utilizan dos tipos de señal, una onda seno y una onda triangular, como se muestra en la Figura 19. Para visualizar las señales análogas en el módulo transmisor y en el módulo receptor se utiliza un Osciloscopio.

8.4 COMPOSICION INTERNA DE LOS MODULOS

Las etapas de transmisión y recepción de cada uno de los sistemas de modulación están compuestas por dos partes principales: Hardware y Software.

8.4.1 Hardware. Esta es la parte física y tangible de cada uno de los módulos y su composición depende del tipo de señal que se va a modular (Digital o Análoga).

Los módulos que realizan modulación de señales digitales están compuestos por, (Ver Figura 20):

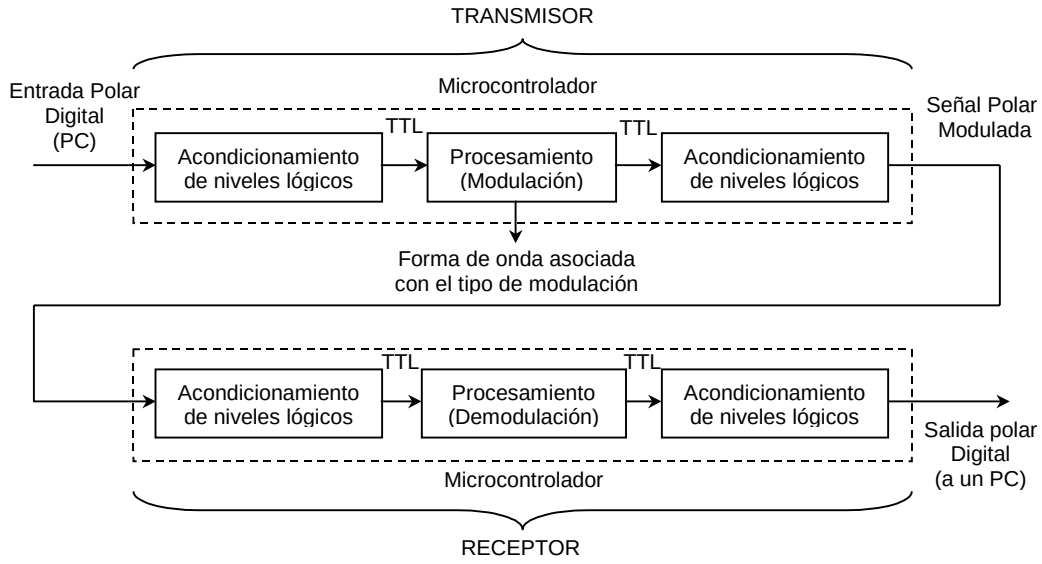


Figura 20 *Esquema general del Hardware de modulación de señales digitales*

- *Acondicionamiento de niveles lógicos*, que consiste en convertir los niveles de voltaje que son enviados o recibidos por un Computador y que representan los estados lógicos, a niveles de voltaje TTL (*Transistor-Transistor Logic*), los cuales son utilizados por el módulo para interpretar los estados lógicos. También se acondiciona la señal que se envía al canal de comunicación, esta se convierte a una señal polar.

- *Procesamiento*, el cual consiste en modular o demodular la señal recibida, este procesamiento es realizado por un microcontrolador que es un Computador completo con prestaciones limitadas destinado a realizar

una tarea asignada, este también se encarga de generar las formas de onda asociadas con cada tipo de modulación.

Los módulos que realizan modulación de señales análogas están compuestos por, (Ver Figura 21):

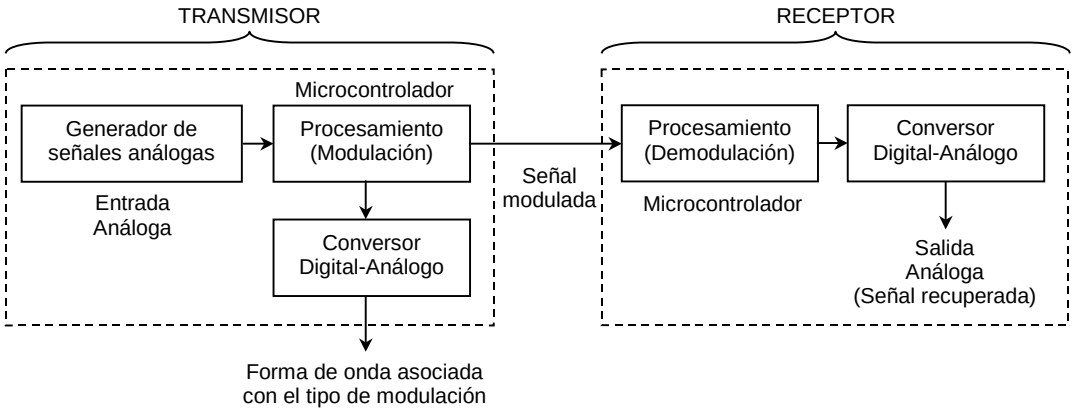


Figura 21 *Esquema general del Hardware de modulación de señales análogas*

- *Generador de señales análogas*, este produce las formas de onda que se utilizaran como señal de entrada para la modulación.

- *Procesamiento,* el cual consiste en modular o demodular la señal recibida, este procesamiento es realizado por un *microcontrolador,* quien además genera las formas de onda asociadas con cada tipo de modulación.

- *Conversión de Digital a Análogo,* este es usado para interpretar la información digital como una señal análoga, que puede corresponder a la señal análoga recuperada o a una forma de oda asociada al tipo de modulación o demodulación.

8.4.2 Software. El software es el programa que reside en la memoria del microcontrolador y esta destinado a gobernar una aplicación determinada, este software esta conformado por las instrucciones que el microcontrolador ejecuta para realizar el proceso de modulación o demodulación y que da como resultado la señal de salida en cada uno de los módulos.

8.5 SELECCION DE COMPONENTES

La selección de los componentes se hizo tomando en cuenta los requerimientos de cada uno de los módulos.

El principal componente de los módulos es el microcontrolador de la familia PIC (*Microchip Technology Inc*.) que posee entre otras características, su Arquitectura RISC (*Reduced Instructions Set Computer*) por lo cual se dispone de un reducido conjunto de instrucciones, Tecnología CMOS de bajo consumo y alta velocidad, además de su amplia versatilidad, como la de tener periféricos incorporados como conversores, comparadores y USART entre otros.

Los microcontroladores utilizados en los módulos son los siguientes:

1. El *PIC16C622*, este se eligió para los sistemas de modulación de señales digitales, principalmente por su velocidad de operación de 20MHz, facilitando así una alta frecuencia de muestreo de una señal de entrada.

2. Ante la decisión de reducir la cantidad de posibles componentes en el diseño de los sistemas de modulación de señales análogas, se elige el microcontrolador *PIC16F874* por poseer un Conversor análogo-digital, una USART y una velocidad de operación de 20MHz, que resultan ser buenas características a usar en este tipo de modulación.

Con el fin de producir una señal que sirva como entrada para los módulos que realizan la modulación de señales análogas se usa el generador de funciones XR2206 de la empresa EXAR, este es capas de generar tres formas de onda diferentes (sinusoidal, triangular y cuadrada).

En los casos en que se necesita convertir una palabra digital a su equivalente nivel de voltaje análogo, se utiliza el conversor digital-análogo DAC0832, el cual tiene una resolución de ocho bits, una alta velocidad de conversión (1μs), interfase directa a un microcontrolador, y trabaja con una fuente sencilla.

8.6 PRINCIPIOS DE DISEÑO PARA LOS MODULOS MANCHESTER, FSK, PSK Y ASK

La transmisión y recepción de cada uno de los módulos utiliza como programa principal un bucle infinito, este es interrumpido por el Timer para obtener una muestra de la línea de entrada, la cual es procesada para actualizar el estado de la línea de transmisión entre transmisor y receptor (ver Figura 22).

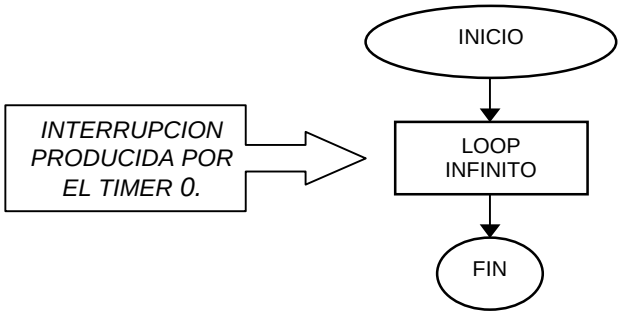


Figura 22 *Programa principal de los módulos Manchester, FSK, PSK y ASK*

Se eligió una tasa de transferencia de bits entre los módulos de 1200Baudios, cuyo periodo de bit es de:

$$T = \frac{1}{f\,(Baudios)} = \frac{1}{1200} = 833{,}33\mu s$$

donde *T* es el periodo de cada bit y *f* es el numero de bits por segundo, siendo este periodo el punto inicial que determina el diseño.

Ya que se desea prevenir errores en la lectura del bit de entrada causados por ruidos u otros factores, se decide entonces, tomar 20 muestras por periodo de bit:

$$T_S = \frac{T}{20} = \frac{833{,}33\mu s}{20} = 41{,}6\mu s$$

donde T_S es el periodo de la frecuencia de muestreo. Es decir, se tomara una muestra cada 41.6µs. Teniendo en cuenta la mínima frecuencia de muestreo que según el teorema de muestreo debe ser:

$$f_S \geq 2\times f\,(bps) \Rightarrow 2\times 1200 = 2400mps$$

donde f_S es la frecuencia de muestreo dada en muestras por segundo (mps), y *f* es el numero de bits por segundo. Esto indica que se deben tomar mínimo dos muestras por periodo de bit, y observando el numero de muestras que se ha definido, el error en la lectura del bit de entrada será mínimo.

De las 20 muestras obtenidas por cada periodo se obtiene un promedio de ceros o unos para asegurar que el bit leído a la entrada posea el estado lógico correcto, y así continuar con la codificación correspondiente a ese bit.

Los módulos Manchester, ASK, y PSK toman una muestra cada 41.6µs lo que proporciona una frecuencia de muestreo de:

$$f_S = \frac{1}{T_S} = \frac{1}{41{,}6\mu s} = 24038 mps$$

En el caso del módulo FSK se toman 24 muestras por periodo de bit:

$$T_S = \frac{T}{24} = \frac{833{,}3\mu s}{24} = 34{,}6\mu s$$

lo que indica que se tomara una muestra cada 34.6µs. Ofreciendo de esta manera una frecuencia de muestreo de:

$$f_S = \frac{1}{T_S} = \frac{1}{34{,}6\mu s} = 28901 mps$$

La razón del aumento en la frecuencia de muestreo de este modulo es la variación de la frecuencia en la línea de transmisión entre el modulo transmisor y el receptor, propia de esta técnica.

Las muestras se toman cada vez que se produce la interrupción del Timer en el microcontrolador, simultáneamente, se genera la señal de salida con un atraso de un bit respecto a la entrada, esto debido al tiempo de procesamiento de las muestras que es exactamente un periodo de bit en cada modulo.

Ya que la señal de entrada proviene del puerto serial de un Computador, el protocolo utilizado es el RS-232, este tipo de comunicación es asincrónica, es decir, los datos no son enviados junto con una señal de sincronismo. Cada palabra es sincronizada utilizando un bit de inicio y un bit de parada, de esta manera, una transmisión comienza con un bit de inicio (0 lógico), luego cada uno de los 8 bits de datos es enviado a través de la línea, uno a la vez; comenzando por el bit menos significativo (LSB).

Un bit de parada (1 lógico) es añadido a la señal para completar la transmisión. Esto se puede apreciar en la Figura 23.

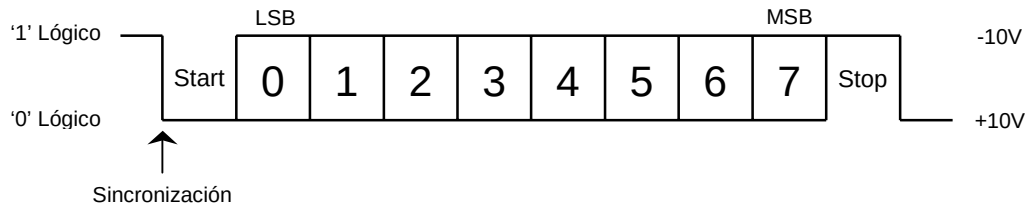


Figura 23 *Forma de onda asincrónica RS-232*

Cuando no hay transmisión de datos, la línea permanece en estado de reposo (1 lógico).

Los módulos aprovechan el estado de reposo para detectar el bit de inicio de una trama, ya que el cambio de estado lógico lo determinara. La detección del bit de inicio no se realiza por una interrupción de flancos de bajada, como habría de esperarse, si no que se utiliza la técnica de *polling* que consiste en observar la línea para determinar si ha ocurrido algún cambio de estado. La principal razón para utilizar este método es que el microcontrolador PIC solo posee un vector de interrupción, en el cual deben estar todas las rutinas de

servicio, esto puede generar conflictos al presentarse dos interrupciones al tiempo, en cuyo caso existiría una prioridad sobre estas, pero esto causaría perdida de información y una inadecuada sincronización. Debido a esto, se decide trabajar solo con la interrupción del Timer.

Cada bit de inicio es utilizado como sincronización en los módulos, para realizar un correcto procesamiento del dato muestreado y así mismo generar su correspondiente señal de salida.

Para prevenir la transmisión o recepción incorrecta de datos al encender el modulo, ya que previamente puede haberse iniciado una transmisión en el Computador, cada módulo cuenta con un tiempo de espera de aproximadamente 150ms que se inicia al colocar en funcionamiento el modulo, ocasionando la interrupción de la misma, lo que a su vez provoca la retransmisión de la información perdida. Si este requiere más tiempo para finalizarla, el tiempo de espera es extendido. Durante esta etapa la línea de transmisión entre el módulo transmisor y el receptor se encuentra en estado de reposo (1 lógico). Al finalizar este tiempo los módulos quedan a la espera de una trama de datos entrante.

8.7 MODULO MANCHESTER

Este módulo transmitirá los bits de una entrada digital utilizando señalización Manchester, este tipo de codificación de línea se caracteriza por generar una transición de estado a la mitad del periodo de un bit.

Para la conversión a Manchester primero se realiza un muestreo de la línea de entrada, esta operación es realizada con ayuda del Timer 0 que se encarga de generar una interrupción a intervalos de tiempo constantes (41,6μs) y durante la cual se toma una muestra del estado de la línea, que es la que contiene la información digital en forma serial.

Al acumular veinte (20) muestras se inicia una operación de procesamiento de las mismas, es decir, se define el estado lógico que tuvo la línea, se realiza la conversión correspondiente al bit que haya sido leído y se identifica si se presentó el inicio o el final de una trama (*Start bit* o *Stop bit*).

Para generar la señal de salida correspondiente al bit codificado se hace también uso de la interrupción del Timer 0, ya que este también tiene la función de actualizar el estado de la línea de transmisión entre los módulos, esta actualización se hace en base al contenido de un BUFFER, el cual es un registro donde se almacenan los cambios de estado que se deben hacer sobre el bit que será transmitido.

La señal codificada a la salida del transmisor deberá tener una transición a mitad de bit, así que teniendo en cuenta que veinte interrupciones completan el periodo de un bit (832µs), se hará el cambio de estado lógico para los bits codificados cada diez interrupciones (416µs), esto se puede apreciar en la Figura 24.

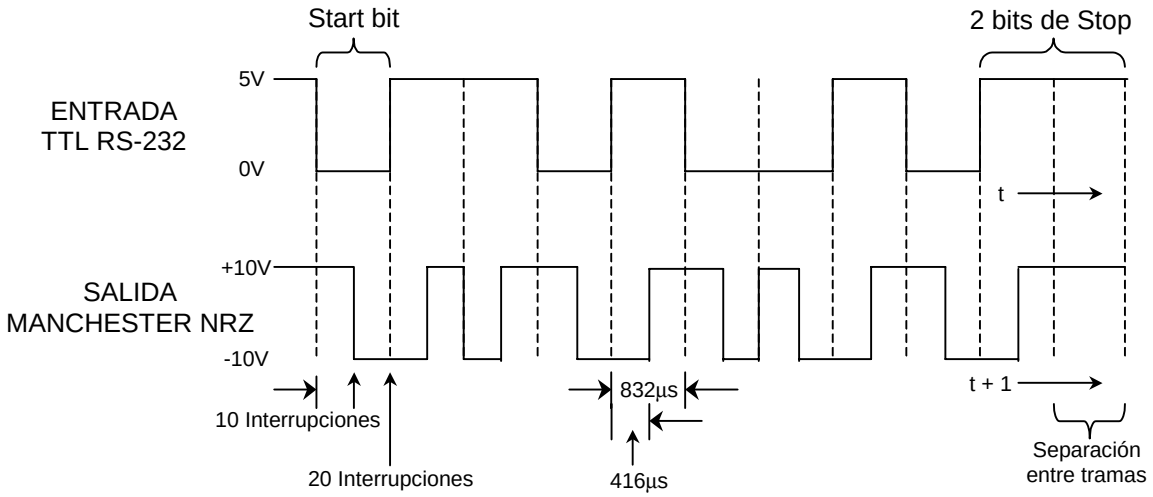


Figura 24 *Formas de onda de entrada RS-232 y salida Manchester*

Como se puede ver en la gráfica un cero lógico es convertido a una transición de un estado alto a uno bajo y un uno lógico es convertido a una transición de un estado bajo a uno alto. Se toma esta forma de onda como convención para representar la codificación Manchester, pero esta convención puede estar de acuerdo o no con la bibliografía existente sobre comunicaciones.

La salida Manchester del módulo esta atrasada un periodo de bit (832µs), con respecto a la señal de entrada, esto se debe al tiempo de procesamiento de las veinte muestras.

La trama de datos utiliza dos bits de parada (*Stop bit*), debido a que se decidió dejar una separación entre tramas de mínimo un periodo de bit para lo cual se usa el segundo bit de parada, esto le permite al módulo detectar el bit de inicio de una próxima trama. De los dos bits sólo el primero se convierte a Manchester, el segundo tendrá un estado lógico uno.

En el módulo de recepción, se decodifica la señal Manchester y se recupera de nuevo la trama de datos.

Para realizar la decodificación se aprovecha que existe un flanco de subida en la señal Manchester que representa un cero lógico y un flanco un flanco de bajada que representa un uno lógico, de esta forma el receptor funciona como un detector de flancos para determinar el bit que se debe recuperar a la salida.

El proceso de transmisión y recepción usado por el módulo Manchester puede ser descrito entonces como se aprecia en la Figura 25:

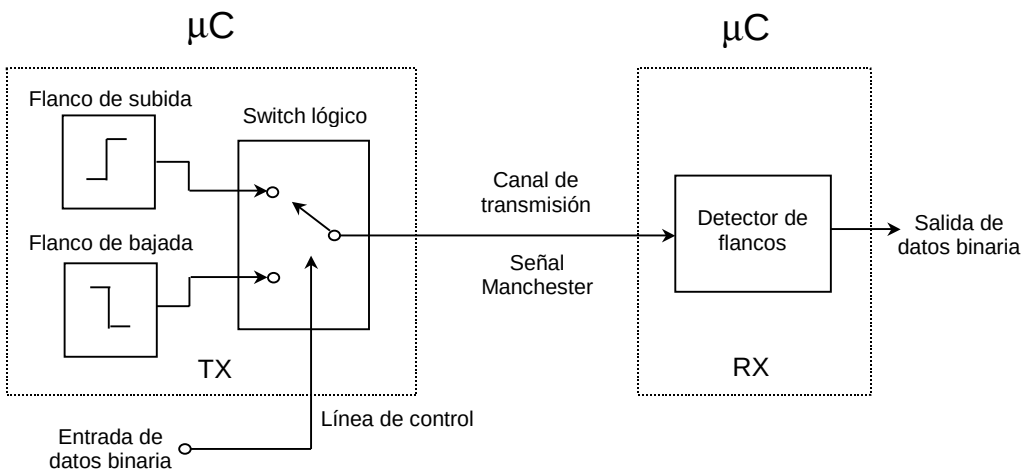


Figura 25 *Diagrama de bloques de los módulos de transmisión y recepción Manchester*

donde el transmisor conmuta la línea de salida entre dos diferentes osciladores, cuya fase esta desplazada de tal manera que se produzca una transición que coincida con la mitad del periodo de bit para así generar los correspondientes flancos, la conmutación se hace de acuerdo a la entrada de datos binaria.

En la etapa de recepción, el detector de flancos se encarga de recuperar los datos binarios, estos son enviados a la salida de acuerdo al flanco que se presente.

8.7.1 Software del transmisor. El software del transmisor es quien realiza la codificación Manchester de la entrada digital.

La información serial de la entrada digital, la cual es entregada por un Computador personal, es llevada al microcontrolador por medio de un driver de línea (MAX232), este se encarga de llevar la señal a niveles de voltaje TTL, correspondientes a los niveles que maneja el microcontrolador.

Cuando se da inicio al programa principal se setea el bit seis del puerto B (RB6), el cual entrega la señal de salida codificada, esto se hace para indicar que la salida se encuentra en estado de reposo. Luego el programa ingresa a un loop infinito, el cual solo es interrumpido por el desbordamiento del Timer 0 que genera una interrupción. Esta interrupción dirige el programa a la dirección de memoria 0004H donde se encuentra el vector de interrupciones. En este sitio se ubica un salto que apunta a la dirección donde se encuentra la rutina de

servicio a la interrupción. Es esta rutina la que finalmente se encarga del procesamiento de la entrada digital (bit siete del puerto B).

La rutina de servicio a la interrupción es la encargada de realizar el muestreo de la entrada digital y de generar la señal de salida. Al iniciarse esta rutina se entra a la subrutina Tiempo de espera, la cual realiza un retardo de aproximadamente 150ms. Para continuar con el programa se debe asegurar que la línea de entrada debe estar en estado de reposo (uno lógico) durante el tiempo de espera, lo contrario indicara que al momento de iniciarse el programa el Computador estaba realizando una transmisión y esta será descartada. Así que la subrutina Tiempo de espera permite que no existan transmisiones al inicio del programa.

Después de ejecutarse el tiempo de espera el programa queda listo para recibir el *Start bit* (cero lógico) de una trama. Al instante en que se detecta un *Start bit* se empieza a contar veinte interrupciones, durante las cuales se lee el estado lógico de la entrada, al completarse estas veinte interrupciones (duración de un bit) se entra a la subrutina Txout donde se verifica el estado lógico del bit leído y se decide el estado con el cual debe iniciar el bit codificado. Este proceso de conteo de veinte interrupciones se repite con los ocho bits de datos y el *Stop bit*. Posterior a la verificación hecha por la subrutina Txout, se realiza un conteo de diez interrupciones después de las cuales se genera la transición de estado a mitad del bit codificado.

Al terminar el muestreo y la codificación de la trama completa, es decir, el *Start bit*, los ocho bits de datos y el *Stop bit*, se verifica que el ultimo bit leído sea un *Stop bit* valido (uno lógico), si es así se realiza un reinicio del programa, es decir, se vuelve a las condiciones de espera de un próximo *Start bit*, pero en el caso en que el ultimo bit leído sea un cero lógico, significara que hay un error de transmisión, por lo cual se dará un aviso de error y se entrara de nuevo a la subrutina Tiempo de espera.

A continuación se hace una descripción detallada de las rutinas implementadas para el desarrollo del codificador Manchester.

8.7.1.1 Programa principal. En esta parte del programa se configura el Timer 0, el cual utiliza la fuente de reloj interna del microcontrolador, se programan el bit siete del puerto B como entrada y los bits cero a seis del puerto B como salida, se inicializan todos los registros que se utilizan en el programa, se habilita la interrupción del Timer 0 y se setea el bit seis del puerto B para indicar que la salida se encuentra en estado de reposo. Esto es realizado únicamente al inicio del programa y no se vuelve a entrar en el mientras el programa este corriendo. Luego el programa queda atrapado en un loop infinito, es decir, se ejecuta un salto que apunta a la dirección de memoria donde se encuentra el mismo salto, este lazo cerrado que se produce solo se interrumpe por la acción de desbordamiento del Timer 0 cada 41,6μs para procesar la entrada digital.

8.7.1.2 Rutina de servicio a la interrupción del Timer 0. Esta rutina inicialmente ejecuta un PUSH, el cual se encarga de almacenar el registro de trabajo W y las banderas de estado de la ALU (Unidad Aritmética Lógica), como Carry (C), Digit Carry (DC) y Cero (Z) que se encuentran en el registro STATUS. Luego se realiza el ajuste de tiempo del Timer 0, este ajuste se realiza debido a que en el momento preciso en que se produce la interrupción, el contador de programa no apunta al vector de interrupción, este solo lo hace hasta cuando se haya ejecutado la instrucción que se estaba efectuando y se debe tener en cuenta que ejecutar una instrucción toma un ciclo de maquina (cuatro periodos de reloj), pero también hay instrucciones que toman dos ciclos de maquina para ser ejecutadas, por lo cual el tiempo en que se produce la interrupción se puede aumentar en dos ciclos de maquina, esto no seria tan notorio si el tiempo a contabilizar fuera del orden de los milisegundos, pero en este caso se contabiliza un tiempo del orden de los microsegundos y se requiere una buena exactitud al contabilizarlo. Con un reloj a 20MHz cada ciclo de maquina es de 200ns, lo cual implica que la interrupción del Timer 0 puede variar entre 41,6 y 42µs. La forma de corregir esto es desplazando la carga del Timer 0 dos ciclos de maquina cuando no exista modificación y quitándolos cuando se presente la modificación, esto se logra capturando el valor del Timer 0 siempre en el mismo sitio, así se sabrá que valor tiene el Timer 0 si no ha sufrido modificación alguna, en este caso el valor capturado es par y si sufre alguna modificación solo cambia un bit, el menos significativo, convirtiendo el valor en impar. De esta manera si el bit menos significativo al realizar la captura del Timer 0 es uno indicara que hubo modificación entonces se procede con la carga del Timer 0, pero si es cero se ejecuta una instrucción de salto, que tarda

dos ciclos de maquina, apuntando a la carga del Timer 0, de esta manera se agregan dos ciclos de maquina cuando el Timer 0 no ha sido modificado. La carga del Timer 0 consiste en calibrar el tiempo en que se producirá la próxima interrupción.

Posteriormente se realiza una actualización del puerto B que consiste en mandar a las salidas del puerto los cambios que se hayan hecho al registro BUFFER. Después se verifica que la bandera TIEMPO sea uno, esto indica que el tiempo de espera ya ha sido ejecutado, en el caso en que sea cero se salta a la subrutina Tiempo de espera. El registro REGCON0 es el registro encargado de controlar la sincronización del programa con el *Start bit* de una trama, si este registro es uno indicara que ya se presento un *Start bit* por lo tanto se debe empezar el conteo de las veinte interrupciones, este conteo lo realiza el registro REG1.

Al completarse veinte interrupciones se llama a la subrutina Txout, esta se encarga de verificar el estado lógico del bit leído y decidir con que estado debe iniciar el bit codificado. El registro REG2 es quien lleva el conteo del numero de bits leídos, si este es diferente de cero indicara que ya se ha leído el primer bit, por lo tanto se debe empezar con la codificación, de esta manera se cuentan diez interrupciones, al final de las cuales se llama a la subrutina Halfbit, esta se encarga de generar la transición de estado del bit codificado a la mitad del bit.

Al realizar la lectura del bit siete (RD) del puerto B, se verifica el estado lógico de la entrada digital, si es un cero lógico se llama a la subrutina Control, la cual

se encarga de llevar el conteo del numero de ceros lógicos que se han leído y de controlar la sincronización del programa con el *Start bit*.

Si el registro REG2 es igual a diez quiere decir que ya se ha leído la trama completa, por lo que se llama a la subrutina Stop, que se encarga de verificar que el ultimo bit leído sea un *Stop bit* valido (uno lógico). Al haberse completado la lectura de la trama completa se produce un reinicio del programa, es decir, se prepara al programa para detectar un próximo *Start bit*, pero este reinicio se ejecuta diez interrupciones después de que se ha leído el *Stop bit*, debido a que se debe permitir que este ultimo termine de ser codificado, ya que la codificación se efectúa después de realizar la lectura.

Por ultimo se ejecuta un POP para restaurar el registro W y las banderas de Carry, Digit Carry y Cero que se habían almacenado a su registro original (STATUS).

8.7.1.3 Subrutina Tiempo de espera. La subrutina de tiempo de espera consiste en un retardo de 150ms, esto se logra contando las interrupciones del Timer 0 que se producen cada 41,6μs, de esta manera se realiza el conteo de 256 interrupciones lo que proporciona un tiempo de 10,7ms, para lograr un tiempo de 150ms se cuentan 14 ciclos de 10,7ms.

Mientras se ejecuta el tiempo de espera se realiza una lectura de la entrada digital, si el resultado de la lectura es un uno lógico (estado de reposo) se sigue

con el conteo hasta completar el tiempo de espera requerido, después de cumplido esto el registro TIEMPO, el cual actúa como bandera para informar que el tiempo de espera ha sido ejecutado, se carga con un uno y la bandera de ERROR (bit cinco del puerto B) se clarea, pero si la lectura da como resultado un cero lógico indicara que una transmisión ha comenzado antes de dar inicio al programa y esto provocara un error de transmisión, lo cual hace que se reinicie de nuevo el tiempo de espera y se setea la bandera de ERROR.

8.7.1.4 Subrutina Txout. Esta subrutina revisa el registro REG0, el cual contiene el numero de lecturas que correspondieron a un cero lógico durante las veinte interrupciones, si de las veinte lecturas realizadas once o más fueron un cero lógico indicara que el bit leído es un cero, pero si de las veinte lecturas tomadas diez o menos fueron un cero lógico quiere decir que el bit leído es un uno. De esta manera se realiza un promedio de las lecturas para descartar los posibles cambios en la señal debidos al ruido que se pueda presentar.

Si el bit leído es un cero, el bit codificado deberá tener un estado inicial alto (uno lógico), pero si el bit leído es un uno, el estado inicial del bit codificado deberá ser bajo (cero lógico). El estado que debe enviarse a la salida es puesto en el bit seis (TX) del registro BUFFER y también es almacenado en el registro REGAUX. Luego se prepara al programa para leer el próximo bit.

8.7.1.5 Subrutina Halfbit. La subrutina Halfbit se encarga de realizar la transición de estado después de haber sido contabilizadas diez interrupciones, es decir, en la mitad del bit que esta siendo codificado. Esto se logra realizando una operación XOR entre el bit seis (TX) del registro BUFFER y un uno lógico, lo que en consecuencia da como resultado el cambio de estado lógico de TX.

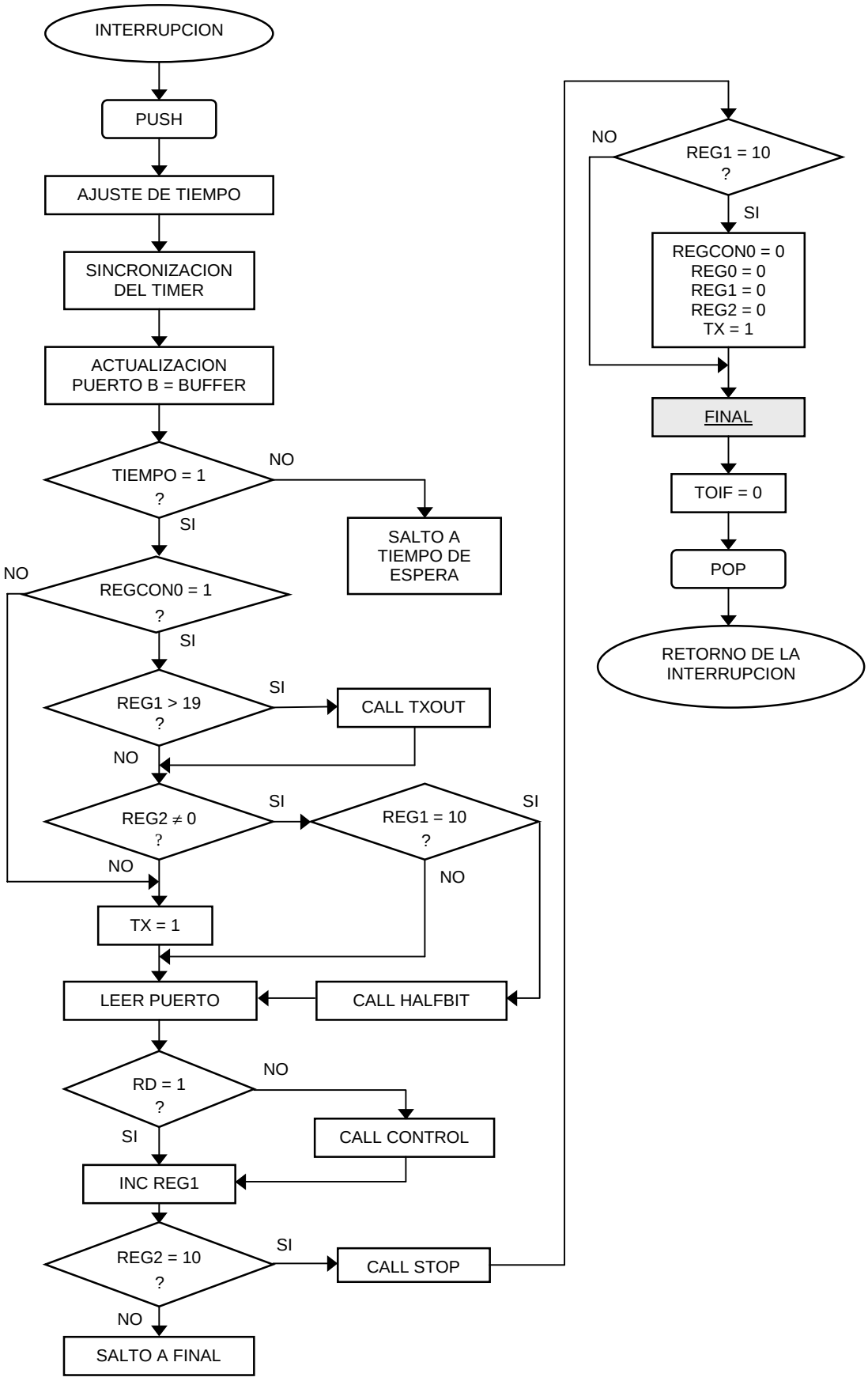
8.7.1.6 Subrutina Control. Esta subrutina es quien realiza el conteo de las lecturas que corresponden a un cero lógico, incrementando el registro REG0. También realiza la sincronización del programa con el *Start bit* de una trama, esta sincronización consiste en preparar al programa para empezar la lectura de todos los bits de la trama, esto se logra inicializando los registros encargados de este proceso. La subrutina Control utiliza el registro REGCON0 como bandera de control para indicar que la sincronización ya se efectuó, la sincronización solo se hace cuando se detecta el *Start bit* y no se realiza mientras se lee la trama completa, después de que esta ha sido leída y codificada el programa queda a la espera de un nuevo *Start bit* y de una nueva sincronización.

8.7.1.7 Subrutina Stop. Esta subrutina consiste en verificar si el ultimo bit leído (décimo bit) corresponde a un *Stop bit* valido, es decir, un uno lógico. La verificación se realiza revisando el contenido del registro REGAUX, el cual almacena el estado inicial del ultimo bit codificado. El estado inicial de un *Stop bit* valido es un cero, ya que su codificación es una transición de un estado bajo a uno alto, de esta manera si el registro REGAUX es igual a uno es por que el

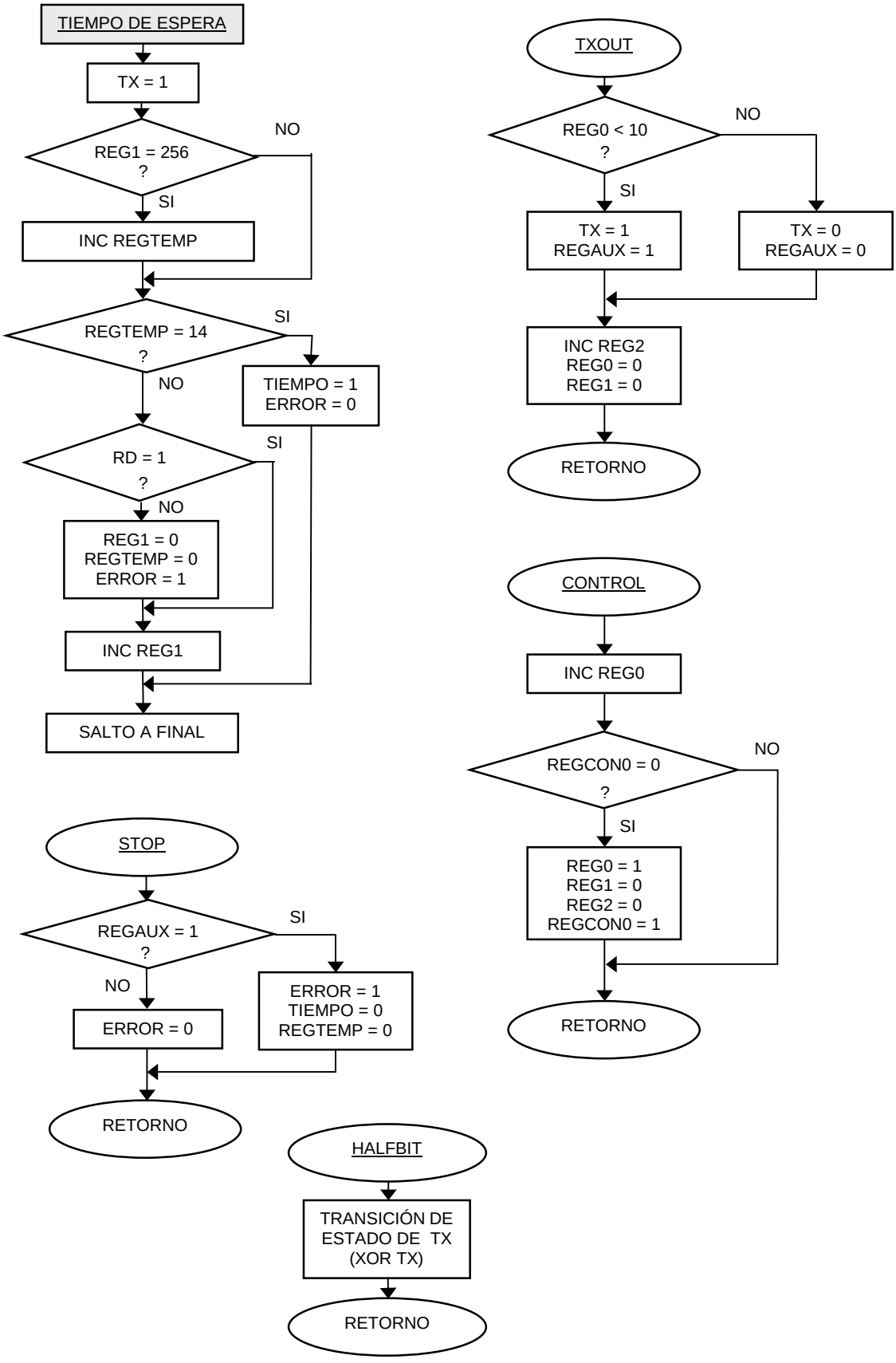
Stop bit no es un uno lógico, por lo que se produce un error de comunicación, para lo cual se setea la bandera de ERROR y se clarea la bandera de TIEMPO obligando al programa a entrar de nuevo a la subrutina Tiempo de espera. En el caso de que el *Stop bit* sea valido se clarea la bandera de ERROR y se retorna.

8.7.2 Diagrama de flujo del transmisor. A continuación se presenta el diagrama de flujo de la rutina de servicio a la interrupción y de las subrutinas que fueron necesarias implementar en el desarrollo de la etapa de transmisión para el módulo Manchester.

RUTINA DE SERVICIO A LA INTERRUPCION PARA EL TRANSMISOR MANCHESTER



SUBRUTINAS


TIEMPO DE ESPERA
TX = 1
REG1 = 256 ?
NO
SI
INC REGTEMP
REGTEMP = 14 ?
SI
NO
TIEMPO = 1
ERROR = 0
RD = 1 ?
SI
NO
REG1 = 0
REGTEMP = 0
ERROR = 1
INC REG1
SALTO A FINAL
TXOUT
REG0 < 10 ?
NO
SI
TX = 1
REGAUX = 1
TX = 0
REGAUX = 0
INC REG2
REG0 = 0
REG1 = 0
RETORNO
CONTROL
INC REG0
REGCON0 = 0 ?
NO
SI
REG0 = 1
REG1 = 0
REG2 = 0
REGCON0 = 1
RETORNO
STOP
REGAUX = 1 ?
SI
NO
ERROR = 1
TIEMPO = 0
REGTEMP = 0
ERROR = 0
RETORNO
HALFBIT
TRANSICIÓN DE ESTADO DE TX (XOR TX)
RETORNO

8.7.3 Programa en código fuente del transmisor. A continuación se da el programa detallado, que se utilizo en el microcontrolador, con sus respectivos comentarios.

```
;-------------------------------------------------------------------------------------------------------------------
;                                   Codificador MANCHESTER (Transmisor)
;-------------------------------------------------------------------------------------------------------------------
        MANCHTX.ASM

        list p=16c622           ;Procesador
        include <p16c622.inc>

;       Al programar el micro tener en cuenta los fusibles de configuración
;       OSC = HS, WDT = OFF, PWRTE = OFF, CP = OFF, BODEN = OFF

;---------------------------------------------------- Zona de Registros ----------------------------------------------


tmr_opt     equ     01h     ;Timer0 / option
status      equ     03h     ;
intcon      equ     0bh     ;
puertob     equ     06h     ;
buffer      equ     2ah     ;Registro que almacena cambios del puertob
reg0        equ     2bh     ;Contador de ceros
reg1        equ     2ch     ;Contador de interrupciones
reg2        equ     2dh     ;Contador de bits
reg3        equ     2eh     ;Contador de las ultimas 10 int. del stop bit
reg4        equ     2fh     ;Contador de interrupciones para osciladores
regaux      equ     30h     ;Registro auxiliar que almacena el contenido de tx
regcon0     equ     31h     ;Bandera de control de start bit
tempwr      equ     32h     ;Temporal del reloj
tiempo      equ     33h     ;Bandera de control del tiempo de espera
regtemp     equ     34h     ;Contador para determinar el tiempo de espera
last        equ     35h     ;Bandera para la señal de sincronía externa
wtemp       equ     70h     ;Temporal de w
stemp       equ     71h     ;Temporal de status


;---------------------------------------------------- Asignaciones de bit ----------------------------------------------


rd          equ     7       ;Entrada serial (RS-232)
tx          equ     6       ;Salida serial (MANCHESTER)
err         equ     5       ;Aviso de error
gie         equ     7       ;Habilita int. general
toie        equ     5       ;Habilita int. por tmr0
toif        equ     2       ;Flag de int. por tmr0
rp0         equ     5       ;Página de memoria
osc1        equ     4       ;Transición de bajada (cero)
osc2        equ     3       ;Transición de subida (uno)
sincro      equ     2       ;Sincronización del osciloscopio
```

```
;------------------------------------------------ Configuración del PIC ----------------------------------------------

        org     0
        goto    inicio          ;Inicio de programa principal

        org     4
        goto    inter           ;Vector de interrupción

        org     5               ;Inicio programa "inicio"

inicio  bsf     status,rp0      ;Va al banco1
        movlw   08h             ;Option, clock interno
        movwf   tmr_opt         ;
        movlw   80h             ;Configuración de puertos
        movwf   puertob         ;Puertob 7:entrada, 6-0:salida
        bcf     status,rp0      ;Retorno al banco0

        clrf    tmr_opt         ;Inicialización de registros
        clrf    puertob         ;
        clrf    reg0            ;
        clrf    reg1            ;
        clrf    reg2            ;
        clrf    reg3            ;
        clrf    reg4            ;
        clrf    regaux          ;
        clrf    regcon0         ;
        clrf    buffer          ;
        clrf    tempwr          ;
        clrf    tiempo          ;
        clrf    regtemp         ;
        clrf    last            ;

        bsf     intcon,gie      ;Habilitación general de interrupciones
        bsf     intcon,toie     ;Habilita interrupción de timer0
        bsf     buffer,tx       ;Inicialización de la salida serial (tx)
        bsf     buffer,osc1     ;Inicialización de señales generadas
        bcf     buffer,osc2     ;
        bcf     buffer,sincro   ;

;------------------------------------------------------- Loop ---------------------------------------------------------

loop    goto    loop            ;Programa principal, loop infinito


;------------------------- RUTINA DE SERVICIO A LA INTERRUPCION DEL TMRO ----------------------

inter   movwf   wtemp           ;Push
        swapf   status,w        ;
        movwf   stemp           ;

        movf    tmr_opt,w       ;Ajuste de tiempo del timer0
        movwf   tempwr          ;
        btfss   tempwr,0        ;
        goto    time3e          ;

time3e  movlw   3eh             ;3eh -> 41,6 useg (aproximadamente)
        movwf   tmr_opt         ;Carga del timer0

princip movf    buffer,w        ;
        movwf   puertob         ;Actualización del puerto b
```

```
        btfss   tiempo,0        ;Se verifica si ya se ejecuto el tiempo de espera
        goto    time            ;
        btfss   regcon0,0       ;Mientras no se de un start bit, tx es 1 (reposo)
        goto    jump            ;

;-------------------------------------------------- Conversión ----------------------------------------------------

        movf    reg1,w          ;
        sublw   13h             ;Reg1 > 19? (Reg1 = 20?)
        btfss   status,0        ;Si reg1 > 19 -> call txout
        call    txout           ;Actualización de tx
        movf    reg2,w          ;Si reg2 es diferente de cero se permite
        btfsc   status,2        ;la conversión del primer bit leído
        goto    jump            ;
        movf    reg1,w          ;
        sublw   0ah             ;Reg1 = 10?
        btfsc   status,2        ;
        call    halfbit         ;Conversión a Manchester (mitad de bit)
        goto    leer

jump    bsf     buffer,tx       ;Estado de reposo de la línea

;-------------------------------------------------- Muestreo ------------------------------------------------------

leer    btfss   puertob,rd      ;Lectura de la entrada serial
        call    control         ;
cont    incf    reg1,f          ;Incrementa contador de interrupciones
        incf    reg4,f          ;Incrementa contador de osciladores

;-------------------------------------------------- Stop ----------------------------------------------------------

stopbit movlw   0ah             ;Verificación del stop (bit 10)
        subwf   reg2,w          ;
        btfss   status,2        ;Reg2 = 10?
        goto    ultimo          ;
        bcf     buffer,err      ;Se asume error=0
        btfss   regaux,tx       ;Chequeo del bit 10 para definir si hay error
        goto    valido          ;Tx = 0 para el bit 10, stop valido
        bsf     buffer,err      ;Tx = 1 para el bit 10, stop no valido
        clrf    tiempo          ;Se inicia de nuevo el tiempo de espera
        clrf    regtemp         ;
        goto    final           ;

;-------------------------------------------------- Reset ---------------------------------------------------------

valido  movlw   0ah             ;El Reset se da en la interrupción 10 después
        subwf   reg1,w          ;que se ha leído el stop bit
        btfss   status,2        ;
        goto    final           ;
        clrf    regcon0         ;Reset
        clrf    reg0            ;
        clrf    reg1            ;
        clrf    reg2            ;
        bsf     buffer,tx       ;
        bsf     last,0          ;Habilita el retorno a cero de la señal de
        goto    final           ;sincronia externa al final del stop bit
```

```
;---------------------------------------------------------- Last ----------------------------------------------------------

ultimo  btfss   last,0          ;
        goto    final           ;

        incf    reg3,f          ;Conteo de las 10 interrupciones faltantes
        movlw   0ah             ;
        subwf   reg3,w          ;
        btfss   status,2        ;Reg3 = 10?
        goto    final           ;

        clrf    last            ;Finalización del stop bit
        clrf    reg3            ;
        movlw   04h             ;Final de la trama, se permite que se vuelva a
        xorwf   buffer,f        ;sincronizar con el osciloscopio (bit 2 del puertob)
        goto    final


;------------------------------------------- Subrutina tiempo de espera -------------------------------------------

time    bsf     buffer,tx       ;Mientras se ejecuta el tiempo de espera, tx es 1 (reposo)
        movlw   0ffh            ;Tiempo de espera de aproximadamente 150mseg
        subwf   reg1,w          ;41.6useg * 256 = 10.6mseg
        btfsc   status,2        ;
        incf    regtemp,f       ;Se cuentan ciclos de 10.6mseg

        movlw   0eh             ;
        subwf   regtemp,w       ;10.6mseg * 14 = 149.1mseg
        btfss   status,2        ;Regtemp = 14 -> tiempo = 1 y error = 0
        goto    again           ;
        bsf     tiempo,0        ;Cumplidos los 150mseg se setea la bandera de tiempo
        bcf     buffer,err      ;Se clarea la bandera de error
        goto    final           ;

again   btfsc   puertob,rd      ;Lectura de la entrada serial
        goto    uno             ;
        clrf    reg1            ;Si rd es cero durante el tiempo de espera
        clrf    regtemp         ;se inicia de nuevo el conteo
        bsf     buffer,err      ;
uno     incf    reg1,f          ;Incremento del contador de interrupciones


;---------------------------------------------------------- Final ----------------------------------------------------------

final   movlw   0ah             ;Generación de las señales que se usan para
        subwf   reg4,w          ;codificar los estados binarios (bits 3 y 4 del puertob)
        btfss   status,2        ;
        goto    salir           ;
        movlw   18h             ;
        xorwf   buffer,f        ;
        clrf    reg4            ;

salir   bcf     intcon,toif     ;Se clarea la bandera del tmr0
        swapf   stemp,w         ;Pop
        movwf   status          ;
        swapf   wtemp,f         ;
        swapf   wtemp,w         ;
        retfie                  ;
```

```
;---------------------------------------------------- Subrutina txout ----------------------------------------------------

txout   movf    reg0,w          ;
        sublw   0ah             ;(10 - reg0)
        btfss   status,0        ;Si reg0 < 10 -> tx=1
        goto    tx0             ;
tx1     bcf     buffer,tx       ;tx=1 (transición de 0 a 1)
        bcf     regaux,tx       ;Se almacena bit leído en regaux
        goto    done            ;

tx0     bsf     buffer,tx       ;tx=0 (transición de 1 a 0)
        bsf     regaux,tx       ;Se almacena bit leído en regaux

done    incf    reg2,f          ;
        clrf    reg0            ;Iniciación de bit
        clrf    reg1            ;
        return                  ;

;---------------------------------------------------- Subrutina halfbit ----------------------------------------------------

halfbit movlw   40h             ;XOR de Tx cada 416useg (10 int.)
        xorwf   buffer,f        ;
        return                  ;

;---------------------------------------------------- Subrutina control ----------------------------------------------------

control incf    reg0,f          ;
        movf    regcon0,w       ;Chequea si regcon0 = 0
        btfsc   status,2        ;
        call    sinc            ;Regcon0 = 0 -> Sincronización con Start bit
        return                  ;

;---------------------------------------------------- Sincronización ----------------------------------------------------

sinc    movlw   04h             ;Señal de sincronización del osciloscopio
        xorwf   buffer,f        ;con el inicio de la trama (bit 2 del puertob)
        clrf    reg0            ;Sincronización con el start bit
        clrf    reg1            ;
        clrf    reg2            ;
        incf    reg0,f          ;
        bsf     regcon0,0       ;Se deshabilita futuras sincronizaciones
        return                  ;por un 0. Se restablece regcon0=0

;-------------------------------------------------------------------------------------------------------------------------
        end
```

8.7.4 Software del receptor. El software del receptor es el encargado de decodificar la señal Manchester y recuperar la información digital. Este software divide el bit codificado en dos segmentos (M1 y M2), cada segmento esta compuesto por diez interrupciones, durante las cuales se realiza la lectura de la entrada codificada, al final de las diez interrupciones se define el estado lógico del segmento leído y se almacena en el registro STORE. Al terminar de leer el bit codificado, en el registro STORE quedara guardada la secuencia de los dos segmentos y esta ayudara a determinar si el bit codificado corresponde a una transición de subida (estado bajo a alto) o a una transición de bajada (estado alto a bajo).

Al inicio del programa principal se setea el bit seis del puerto B (RB6), el cual entrega la señal de salida decodificada, esto se hace para indicar que la salida se encuentra en estado de reposo. Luego el programa ingresa a un loop infinito, el cual solo es interrumpido por el desbordamiento del Timer 0 que genera una interrupción. Esta interrupción dirige el programa a la dirección de memoria 0004H donde se encuentra el vector de interrupciones. En este sitio se ubica un salto que apunta a la dirección donde se encuentra la rutina de servicio a la interrupción. Es esta rutina la que finalmente se encarga del procesamiento de la entrada codificada (bit siete del puerto B).

La rutina de servicio a la interrupción es la encargada de realizar el muestreo de la entrada codificada y de generar la señal de salida. Al iniciarse esta rutina se entra a la subrutina Tiempo de espera, la cual realiza un retardo de

aproximadamente 150ms en el cual se verifica que el Computador no este transmitiendo al momento de iniciar el programa.

Luego de ejecutar el tiempo de espera el programa queda listo para recibir el *Start bit* codificado de una trama. Al instante en que se detecta un *Start bit* se empieza a contar diez interrupciones, durante las cuales se lee el estado lógico de la entrada, al completarse estas diez interrupciones (duración del primer segmento del bit codificado) se entra a la subrutina Chequeo1 donde se verifica el estado lógico del segmento leído y se almacena en el registro STORE, después se continua con el conteo de interrupciones hasta completar veinte, al final de las cuales se llama a la subrutina Chequeo2, esta determina el estado lógico del segundo segmento leído y decide que tipo de transición se presento (subida o bajada), de acuerdo a esto se define el estado lógico de la salida digital. Este proceso es repetido con los ocho bits de datos y el *Stop bit*.

Al terminar el muestreo y la decodificación de la trama completa, se verifica que el ultimo bit leído sea un *Stop bit* valido, si es así se realiza un reinicio del programa, es decir, se vuelve a las condiciones de espera de un próximo *Start bit*, pero en el caso en que el ultimo bit leído sea un cero lógico, significara que hay un error de transmisión, por lo cual se dará un aviso de error y se entrara de nuevo a la subrutina Tiempo de espera.

A continuación se hace una descripción detallada de las rutinas implementadas para el desarrollo del decodificador Manchester.

8.7.4.1 Programa principal. En esta parte del programa se configura el Timer 0, el cual utiliza la fuente de reloj interna del microcontrolador, se programan el bit siete del puerto B como entrada y los bits cero a seis del puerto B como salida, se inicializan todos los registros que se utilizan en el programa, se habilita la interrupción del Timer 0 y se setea el bit seis del puerto B para indicar que la salida se encuentra en estado de reposo. Esto es realizado únicamente al inicio del programa y no se vuelve a entrar en el mientras el programa este corriendo. Luego el programa ingresa a un loop infinito, es decir, se ejecuta un salto que apunta a la dirección de memoria donde se encuentra el mismo salto, este lazo cerrado que se produce solo se interrumpe por la acción de desbordamiento del Timer 0 cada 41,6µs para procesar la entrada digital.

8.7.4.2 Rutina de servicio a la interrupción del Timer 0. Esta rutina inicialmente ejecuta un PUSH, el cual se encarga de almacenar el registro de trabajo W y las banderas de estado de la ALU. Luego se realiza el ajuste de tiempo del Timer 0, este ajuste se realiza para cargar el Timer 0 correctamente y de esta manera calibrar el tiempo en que se presentara la próxima interrupción.

Posteriormente se realiza una actualización del puerto B que consiste en mandar a las salidas del puerto los cambios que se hayan hecho al registro BUFFER. Después se verifica que la bandera TIEMPO sea uno, esto indica que el tiempo de espera ya ha sido ejecutado, en el caso en que sea cero se

salta a la subrutina Tiempo de espera. El registro REGCON0 es el registro encargado de controlar la sincronización del programa con el *Start bit* codificado, si este registro es uno indicara que ya se presento un *Start bit* por lo tanto se debe empezar el conteo de diez interrupciones, este conteo lo realiza el registro REG1.

Al completarse diez interrupciones se llama a la subrutina Chequeo1, esta se encarga de verificar el estado lógico del primer segmento leído, después se continua con el conteo de interrupciones hasta completar veinte, esto hace que se llame a la subrutina Chequeo2, que verifica el estado lógico del segundo segmento leído, determinando de esta manera el tipo de transición que se presento y el estado lógico de la salida digital.

Al realizar la lectura del bit siete (RD) del puerto B, se verifica el estado lógico de la entrada codificada, si es un cero lógico se llama a la subrutina Control, la cual se encarga de llevar el conteo del numero de ceros lógicos que se han leído y de controlar la sincronización del programa con el *Start bit* codificado.

Si el registro REG2 es igual a diez, indicara que se ha leído la trama completa, por lo que se llama a la subrutina Stop, que se encarga de verificar que el ultimo bit leído sea un *Stop bit* valido. Dentro de esta subrutina se produce el reinicio del programa, es decir, se prepara al programa para detectar un próximo *Start bit*.

Por ultimo se ejecuta un POP para restaurar el registro W y las banderas de estado de la ALU a su registro original (STATUS).

8.7.4.3 Subrutina Tiempo de espera. La subrutina de tiempo de espera consiste en un retardo de 150ms, esto se logra contando las interrupciones del Timer 0 que se producen cada 41,6μs, de esta manera se realiza el conteo de 256 interrupciones lo que proporciona un tiempo de 10,7ms, para lograr un tiempo de 150ms se cuentan 14 ciclos de 10,7ms.

Mientras se ejecuta el tiempo de espera se realiza una lectura de la entrada codificada, si el resultado de la lectura es un uno lógico (estado de reposo) se sigue con el conteo hasta completar el tiempo de espera requerido, después de cumplido esto el registro TIEMPO, el cual actúa como bandera para informar que el tiempo de espera ha sido ejecutado, se carga con un uno y la bandera de ERROR (bit cinco del puerto B) se clarea, pero si la lectura da como resultado un cero lógico indicara que una transmisión ha comenzado antes de dar inicio al programa y esto provocara un error de transmisión, lo cual hace que se reinicie de nuevo el tiempo de espera y se setea la bandera de ERROR.

8.7.4.4 Subrutina Chequeo1. Esta subrutina revisa el registro REG0, el cual contiene el numero de lecturas que correspondieron a un cero lógico durante diez interrupciones, si de las diez lecturas realizadas seis o más fueron un cero lógico indicara que el segmento leído es un cero, pero si de las diez lecturas

tomadas cinco o menos fueron un cero lógico quiere decir que el segmento leído es un uno. De esta manera se realiza un promedio de las lecturas para descartar los posibles cambios en la señal debidos al ruido que se pueda presentar.

Si el segmento leído es un cero, se clarea el bit cero (M1) del registro STORE, pero si el segmento leído es un uno, el bit 0 del registro STORE se carga con un uno, de esta manera se almacena el estado lógico del segmento leído.

8.7.4.5 Subrutina Chequeo2. Esta subrutina también revisa el contenido del registro REG0, en el cual se han almacenado las lecturas que correspondieron a un cero lógico durante otras diez interrupciones, si de las diez lecturas realizadas seis o más fueron un cero lógico indicara que el segmento leído es un cero, pero si de las diez lecturas tomadas cinco o menos fueron un cero lógico quiere decir que el segmento leído es un uno. De esta manera se realiza un promedio de las lecturas para descartar posibles cambios en la señal debidos al ruido que se pueda presentar. Si el segundo segmento leído es un cero, el bit uno (M2) del registro STORE es clareado, pero si el segundo segmento es un uno, el bit uno del registro STORE se carga con un uno, de esta manera se almacena el estado lógico del segundo segmento. Luego se verifica el contenido del registro STORE, si este es igual a dos indicara que se presento una transición de subida determinando que el bit decodificado debe tener un estado lógico cero, pero si el registro STORE es igual a uno indicara que se presenta una transición de bajada determinando que el bit decodificado

debe tener un estado lógico uno. El estado que debe enviarse a la salida es puesto en el bit seis (TX) del registro BUFFER. Luego se prepara al programa para leer el próximo bit.

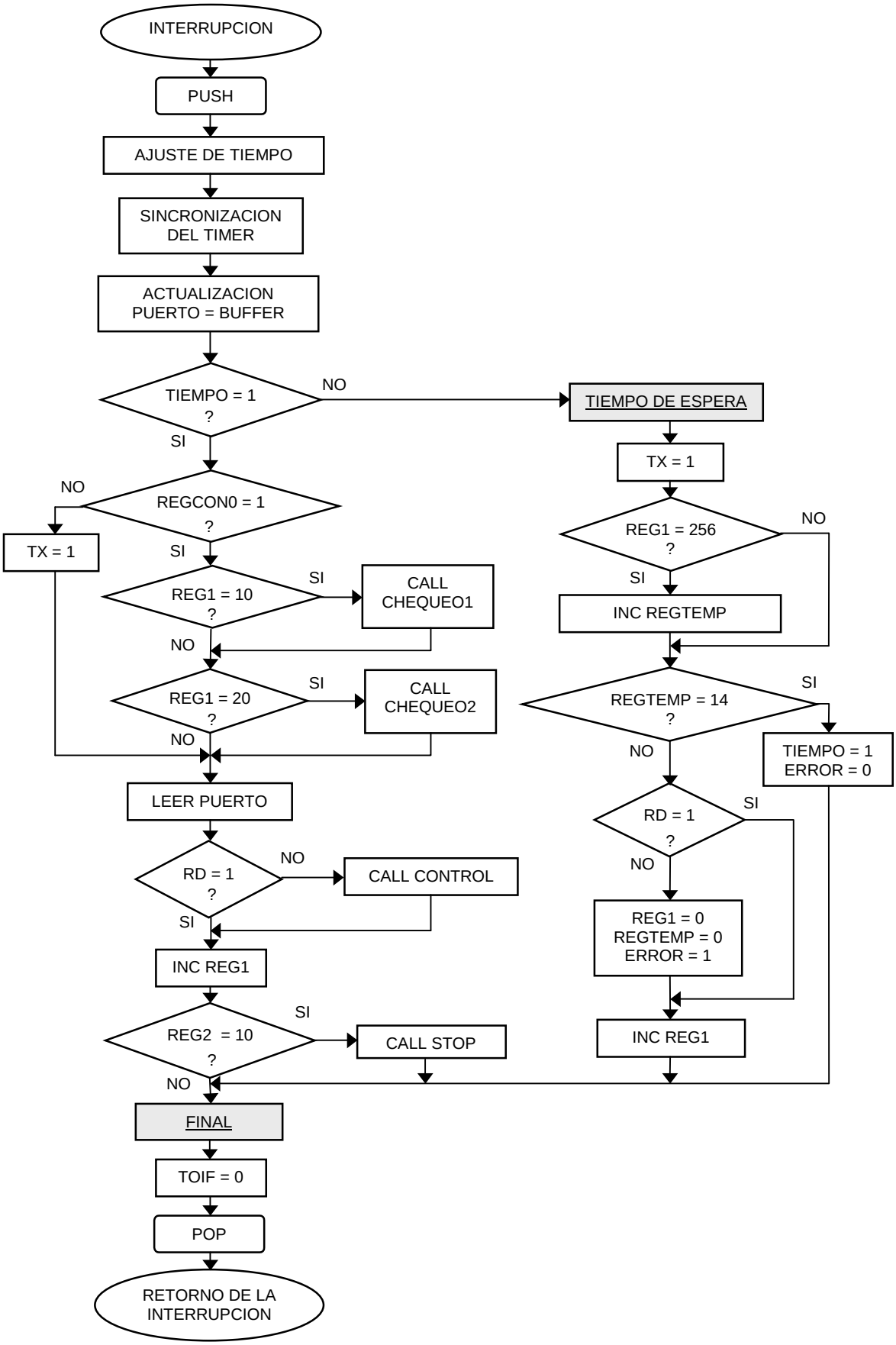
8.7.4.6 Subrutina Control. Esta subrutina es quien realiza el conteo de las lecturas que corresponden a un cero lógico, incrementando el registro REG0. También realiza la sincronización del programa con el *Start bit* codificado de una trama, esta sincronización consiste en preparar al programa para empezar la lectura de todos los bits codificados, esto se logra inicializando los registros encargados de este proceso. La subrutina Control utiliza el registro REGCON0 como bandera de control para indicar que la sincronización ya se efectuó, la sincronización solo se hace cuando se detecta el *Start bit* codificado y no se realiza mientras se lee la trama codificada, después de que esta ha sido leída y decodificada el programa queda a la espera de un nuevo *Start bit* y de una nueva sincronización.

8.7.4.7 Subrutina Stop. Esta subrutina consiste en verificar que el ultimo bit leído (décimo bit) corresponda a un *Stop bit* valido, es decir, un uno lógico. La verificación se realiza revisando el contenido del bit seis (TX) del registro BUFFER, el cual contiene el estado lógico del ultimo bit decodificado. Si el estado de TX es uno indicara que se presento un *Stop bit* valido, pero si TX es cero se produce un error de comunicación, para lo cual se setea la bandera de ERROR y se clarea la bandera de TIEMPO obligando al programa a entrar de

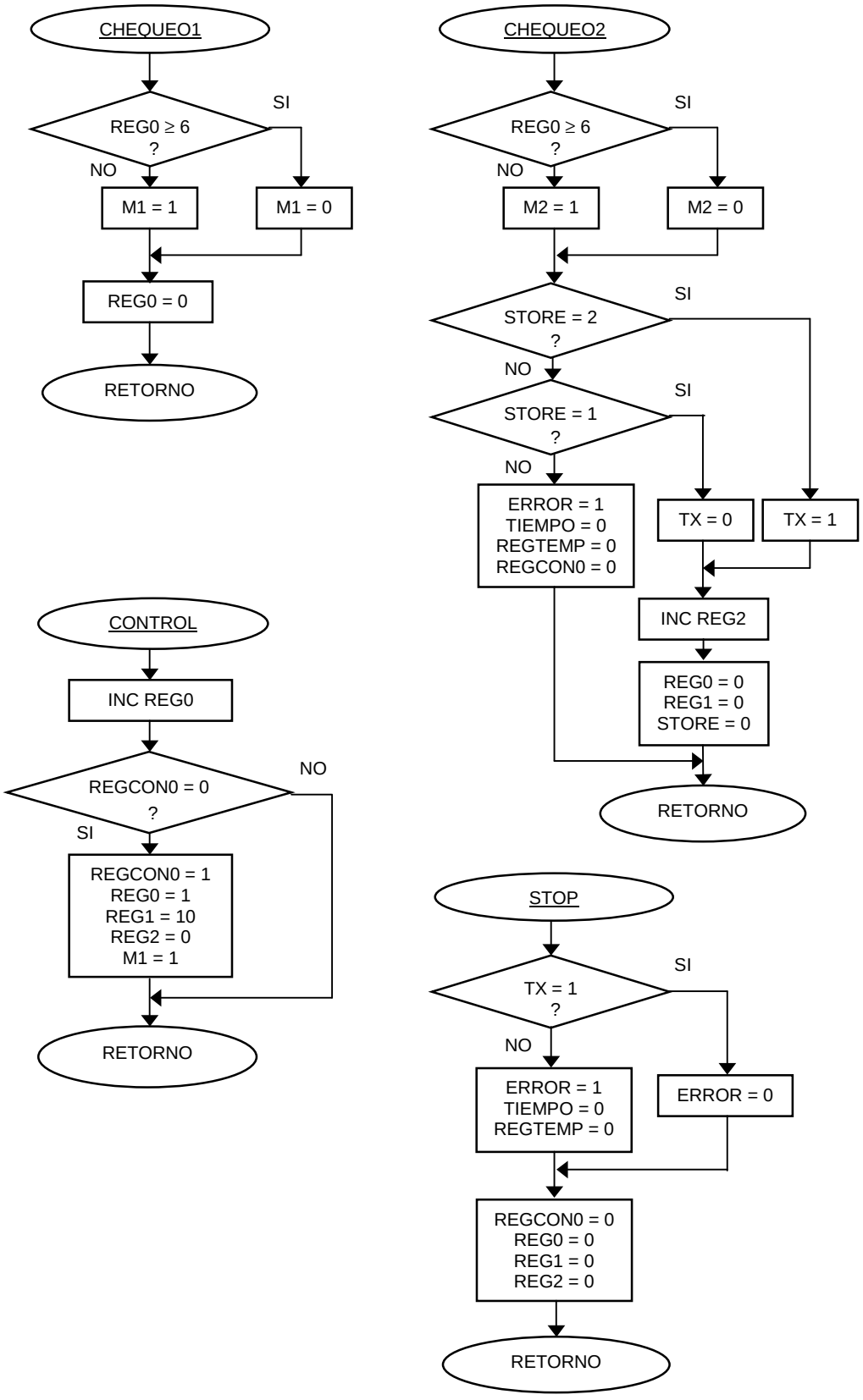
nuevo a la subrutina Tiempo de espera. En el caso de que el *Stop bit* sea valido se clarea la bandera de ERROR y se realiza el reinicio del programa, es decir, se prepara al programa para recibir un nuevo *Start bit* codificado.

8.7.5 Diagrama de flujo del receptor. A continuación se presenta el diagrama de flujo de la rutina de servicio a la interrupción y de las subrutinas que fueron necesarias implementar en el desarrollo de la etapa de recepción para el módulo Manchester.

RUTINA DE SERVICIO A LA INTERRUPCION DEL RECEPTOR MANCHESTER



SUBRUTINAS


CHEQUEO1
REG0 ≥ 6 ?
SI
NO
M1 = 1
M1 = 0
REG0 = 0
RETORNO
CHEQUEO2
REG0 ≥ 6 ?
SI
NO
M2 = 1
M2 = 0
STORE = 2 ?
SI
NO
STORE = 1 ?
SI
NO
ERROR = 1
TIEMPO = 0
REGTEMP = 0
REGCON0 = 0
TX = 0
TX = 1
INC REG2
REG0 = 0
REG1 = 0
STORE = 0
RETORNO
CONTROL
INC REG0
REGCON0 = 0 ?
NO
SI
REGCON0 = 1
REG0 = 1
REG1 = 10
REG2 = 0
M1 = 1
RETORNO
STOP
TX = 1 ?
SI
NO
ERROR = 1
TIEMPO = 0
REGTEMP = 0
ERROR = 0
REGCON0 = 0
REG0 = 0
REG1 = 0
REG2 = 0
RETORNO

8.7.6 Programa en código fuente del receptor. A continuación se da el programa detallado, que se utilizo en el microcontrolador, con sus respectivos comentarios.

```
;-------------------------------------------------------------------------------------------------------------------
;                                   Decodificador MANCHESTER (Receptor)
;-------------------------------------------------------------------------------------------------------------------
;       MANCHRX.ASM

        list p=16c622           ;Procesador
        include <p16c622.inc>

;       Al programar el micro tener en cuenta los fusibles de configuración
;       OSC = HS, WDT = OFF, PWRTE = OFF, CP = OFF, BODEN = OFF


;------------------------------------------------------ Zona de Registros ----------------------------------------------


tmr_opt     equ     01h     ;Timer0 / option
status      equ     03h     ;
intcon      equ     0bh     ;
puertob     equ     06h     ;
buffer      equ     2ah     ;Registro que almacena cambios del puertob
reg0        equ     2bh     ;Contador de ceros
reg1        equ     2ch     ;Contador de interrupciones
reg2        equ     2dh     ;Contador de bits
regcon0     equ     2eh     ;Bandera de control de start bit
tempwr      equ     2fh     ;Temporal del reloj
store       equ     30h     ;Registro que almacena segmentos del bit leído
tiempo      equ     31h     ;Bandera de control del tiempo de espera
regtemp     equ     32h     ;Contador para determinar el tiempo de espera
wtemp       equ     70h     ;Temporal de w
stemp       equ     71h     ;Temporal de status


;------------------------------------------------------ Asignaciones de bit ----------------------------------------


rd          equ     7       ;Entrada serial (MANCHESTER)
tx          equ     6       ;Salida serial (RS-232)
err         equ     5       ;Aviso de error
gie         equ     7       ;Habilita int. general
toie        equ     5       ;Habilita int. por tmr0
toif        equ     2       ;Flag de int. por tmr0
rp0         equ     5       ;Página de memoria
m1          equ     0       ;Segmentos de bit (m1 = primera mitad)
m2          equ     1       ;m2 = segunda mitad
sincro      equ     2       ;Sincronización del osciloscopio
```

```
;---------------------------------------------- Configuración del PIC ----------------------------------------------

        org     0
        goto    inicio          ;Inicio de programa principal

        org     4
        goto    inter           ;Vector de interrupción

        org     5               ;Inicio programa "inicio"

inicio  bsf     status,rp0      ;Va al banco1
        movlw   08h             ;Option, clock interno
        movwf   tmr_opt         ;
        movlw   80h             ;Configuración de puertos
        movwf   puertob         ;Puertob 7:entrada, 6-0:salida
        bcf     status,rp0      ;Retorno al banco0
        clrf    tmr_opt         ;Inicialización de registros
        clrf    puertob         ;
        clrf    reg0            ;
        clrf    reg1            ;
        clrf    reg2            ;
        clrf    regcon0         ;
        clrf    buffer          ;
        clrf    tempwr          ;
        clrf    store           ;
        clrf    tiempo          ;
        clrf    regtemp         ;
        bsf     intcon,gie      ;Habilitación general de interrupciones
        bsf     intcon,toie     ;Habilita interrupción del timer0
        bsf     buffer,tx       ;Inicialización del puerto b

;------------------------------------------------------- Loop ---------------------------------------------------------

loop    goto    loop            ;Programa principal, loop infinito


;------------------------ RUTINA DE SERVICIO A LA INTERRUPCION DEL TMRO ----------------------

inter   movwf   wtemp           ;Push
        swapf   status,w        ;
        movwf   stemp           ;

        movf    tmr_opt,w       ;Ajuste de tiempo del timer0
        movwf   tempwr          ;
        btfss   tempwr,0        ;
        goto    time3e          ;

time3e  movlw   3eh             ;3eh -> 41,6 useg (aproximadamente)
        movwf   tmr_opt         ;Carga del timer0

princip movf    buffer,w        ;
        movwf   puertob         ;Actualización del puerto b

        btfss   tiempo,0        ;Se verifica si ya se ejecuto el tiempo de espera
        goto    time            ;

        btfss   regcon0,0       ;Mientras no se de un start bit, tx es 1 (reposo)
        goto    jump            ;
```

```
;---------------------------------------------------- Conversión ----------------------------------------------------

        movf    reg1,w          ;
        sublw   0ah             ;Reg1 = 10?
        btfsc   status,2        ;Si reg1 = 10 -> call Check1
        call    check1          ;Chequea la primera mitad del bit

        movf    reg1,w          ;
        sublw   14h             ;Reg1 = 20?
        btfsc   status,2        ;Si reg1 = 20  -> call Check2
        call    check2          ;Chequea la segunda mitad del bit y convierte a RS-232
        goto    leer            ;

jump    bsf     buffer,tx       ;Estado de reposo de la línea

;---------------------------------------------------- Muestreo ----------------------------------------------------

leer    btfss   puertob,rd      ;Lectura de la entrada serial
        call    control         ;
        incf    reg1,f          ;Incrementa contador de interrupciones

stopbit movlw   0ah             ;Verificación del stop (bit 10)
        subwf   reg2,w          ;
        btfsc   status,2        ;Reg2 = 10?
        call    stop            ;Chequea si el bit 10 fue un 1
        goto    final           ;

;------------------------------------------- Subrutina tiempo de espera -------------------------------------------

time    bsf     buffer,tx       ;Mientras se ejecuta el tiempo de espera, tx es 1 (reposo)
        movlw   0ffh            ;Tiempo de espera de aproximadamente 150mseg
        subwf   reg1,w          ;41.6useg * 256 = 10.6mseg
        btfsc   status,2        ;
        incf    regtemp,f       ;Se cuentan ciclos de 10.6mseg

        movlw   0eh             ;
        subwf   regtemp,w       ;10.6mseg * 14 = 149.1mseg
        btfss   status,2        ;Regtemp = 14 -> tiempo = 1 y error = 0
        goto    again           ;
        bsf     tiempo,0        ;Cumplidos los 150mseg se setea la bandera de tiempo
        bcf     buffer,err      ;Se clarea la bandera de error
        goto    final           ;

again   btfsc   puertob,rd      ;Lectura de la entrada serial
        goto    uno             ;
        clrf    reg1            ;Si rd es cero durante el tiempo de espera
        clrf    regtemp         ;se inicia de nuevo el conteo
        bsf     buffer,err      ;
uno     incf    reg1,f          ;Incremento del contador de interrupciones

;------------------------------------------------------- Final -------------------------------------------------------

final   bcf     intcon,toif     ;Se clarea la bandera del tmr0
        swapf   stemp,w         ;Pop
        movwf   status          ;
        swapf   wtemp,f         ;
        swapf   wtemp,w         ;
        retfie                  ;
```

```
;------------------------------------------------- Subrutina check1 ------------------------------------------------

check1 movlw  06h          ;Primer Segmento (m1)
       subwf  reg0,w       ;
       bcf    store,m1     ;Se asume m1 = 0
       btfss  status,0     ;Si reg0 >= 6 -> m1 = 0
       bsf    store,m1     ;
       clrf   reg0         ;Inicio de la segunda mitad
       return              ;

;------------------------------------------------- Subrutina check2 ------------------------------------------------

check2 movlw  06h          ;Segundo segmento (m2)
       subwf  reg0,w       ;
       bcf    store,m2     ;Se asume m2 = 0
       btfss  status,0     ;Si reg0 >= 6 -> m2 = 0
       bsf    store,m2     ;
uno    movlw  02h          ;Secuencia 0-1 (m1-m2) para un uno en MANCHESTER
       subwf  store,w      ;
       btfss  status,2     ;
       goto   cero         ;
       bsf    buffer,tx    ;Conversión a RS-232 para un uno
       goto   next

cero   movlw  01h          ;Secuencia 1-0 (m1-m2) para un cero en MANCHESTER
       subwf  store,w      ;
       btfss  status,2     ;
       goto   fail         ;
       bcf    buffer,tx    ;Conversión a RS-232 para un cero
next   incf   reg2,f       ;
       clrf   store        ;Iniciación de bit
       clrf   reg0         ;
       clrf   reg1         ;
       return              ;Retorno

fail   bsf    buffer,err   ;Aviso de error si el bit leído no corresponde
       clrf   tiempo       ;a un uno o un cero en MANCHESTER
       clrf   regtemp      ;
       clrf   regcon0      ;
       return              ;Retorno

;------------------------------------------------- Subrutina stop --------------------------------------------------

stop   bcf    buffer,err   ;Se asume error = 0
       btfsc  buffer,tx    ;Chequeo del bit 10 para definir si hay error
       goto   valido       ;Tx = 1 para el bit 10, stop valido
       bsf    buffer,err   ;Tx = 0 para el bit 10, stop no valido
       clrf   tiempo       ;Se inicia de nuevo el tiempo de espera
       clrf   regtemp      ;
valido clrf   regcon0      ;
       clrf   reg0         ;
       clrf   reg1         ;
       clrf   reg2         ;
       movlw  04h          ;Final de la trama, se permite que se vuelva a
       xorwf  buffer,f     ;sincronizar con el osciloscopio (bit 2 del puertob)
       return              ;
```

```
;------------------------------------------------- Subrutina control ---------------------------------------------

control  incf     reg0,f          ;
         movf     regcon0,w       ;Chequea si regcon0 = 0
         btfsc    status,2        ;
         call     sinc            ;Regcon0 = 0 -> Sincronización con Start bit
         return                   ;

;------------------------------------------------- Sincronización ------------------------------------------------

sinc     movlw    04h             ;Señal de sincronización del osciloscopio
         xorwf    buffer,f        ;con el inicio de la trama (bit 2 del puertob)
         clrf     reg0            ;Sincronización con el start bit
         movlw    0ah             ;Se asume que el primer segmento (m1) ya ha
         movwf    reg1            ;transcurrido (stop bit Manchester)
         bsf      store,m1        ;
         clrf     reg2            ;
         incf     reg0,f          ;
         bsf      regcon0,0       ;Se deshabilitan futuras sincronizaciones
         return                   ;por un 0. Se restablece con el Stop bit

;-------------------------------------------------------------------------------------------------------------------
         end
```

8.8 MODULO FSK

El objetivo de este módulo es transmitir una señal digital utilizando modulación FSK, este tipo de modulación consiste en desplazar la frecuencia de una señal portadora (en este caso una onda cuadrada) desde una frecuencia de marca (para un 1 binario) hasta una frecuencia de espacio (para un 0 binario) de acuerdo con la señal de entrada digital.

El primer paso en la conversión a FSK es realizar un muestreo de la línea de entrada, esta operación es ejecutada por el Timer 0 que se encarga de generar una interrupción a intervalos de tiempo constantes (34,6μs) y durante la cual se toma una muestra del estado de la línea, que es la que contiene la información digital en forma serial.

Al acumularse un total de veinticuatro (24) muestras, es completado un periodo de bit (831μs) y se inicia una operación de procesamiento de las mismas, es decir, se define el estado lógico que tuvo la línea, se realiza la modulación correspondiente al bit que haya sido leído y se identifica si se presentó el inicio o el final de una trama (*Start bit* o *Stop bit*), es decir la posición del bit leído dentro de la trama.

La señal de salida correspondiente al bit modulado hace también uso de la interrupción del Timer 0, ya que este también tiene la función de actualizar el estado de la línea de transmisión entre los módulos, esta actualización se hace en base al contenido de un BUFFER, el cual es un registro donde se

almacenan los cambios de estado que se deben hacer a la línea entre los módulos.

Para generar la señal de salida del transmisor se varia la frecuencia de un tren de pulsos (señal portadora), para lograr esto y teniendo en cuenta que veinticuatro interrupciones completan el periodo de un bit, se decide que para una de las frecuencias se hará cambios de estado lógico cada cuatro interrupciones formando así una onda cuadrada de 2,4Khz y para la otra frecuencia se harán cambios de estado cada seis interrupciones formando una onda cuadrada de 3,6Khz, las formas de onda se pueden apreciar en la Figura 26.

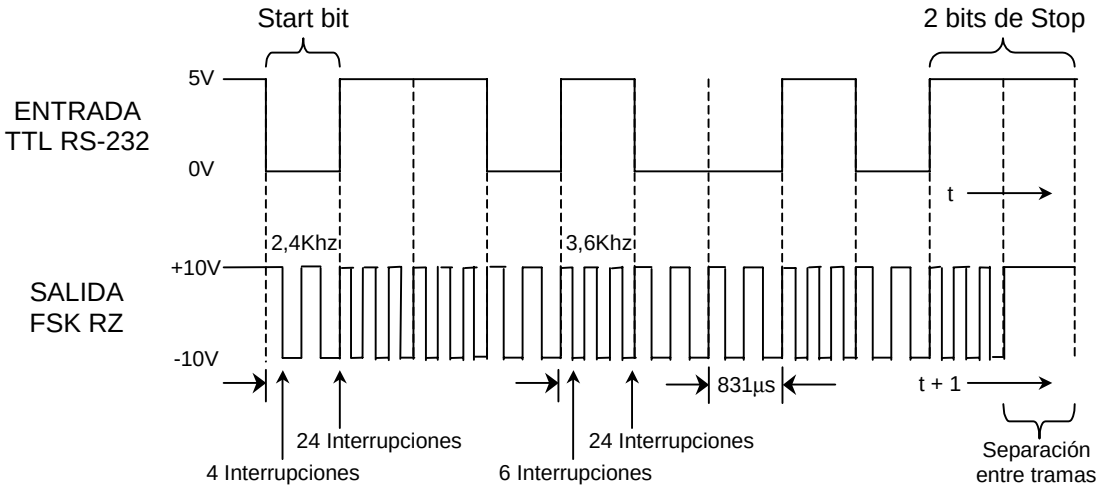


Figura 26 *Formas de onda de entrada RS-232 y salida FSK*

Se observa en la gráfica que un cero lógico es convertido a un tren de pulsos con una frecuencia de 2,4Khz y un uno lógico es convertido a un tren de pulsos con una frecuencia de 3,6Khz. La salida FSK del módulo esta atrasada un periodo de bit (831µs), con respecto a la señal de entrada, esto se debe al tiempo de procesamiento de las veinticuatro muestras.

La trama de datos utiliza dos bits de parada (*Stop bit*), debido a que se decidió dejar una separación entre tramas de mínimo un periodo de bit para lo cual se usa el segundo bit de parada, esto le permite al módulo detectar el bit de inicio de una próxima trama. De los dos bits sólo el primero se convierte a FSK, el segundo tendrá un estado lógico uno.

En el módulo de recepción se demodula la señal FSK y se recupera de nuevo la trama de datos. Para lograr la demodulación se aprovechara que existen dos señales de diferente frecuencia que representan los estados lógicos cero y uno, de esta manera el receptor hará las funciones de un conversor Frecuencia-Voltaje, así la frecuencia más baja provocara una salida de cero voltios (0V) y la frecuencia más alta provocara una salida de cinco voltios (+5V), luego estos voltajes son elevados a los niveles de voltaje adecuados para la salida RS-232.

En resumen el proceso de transmisión y recepción usado por el módulo FSK puede describirse como se aprecia en la Figura 27:

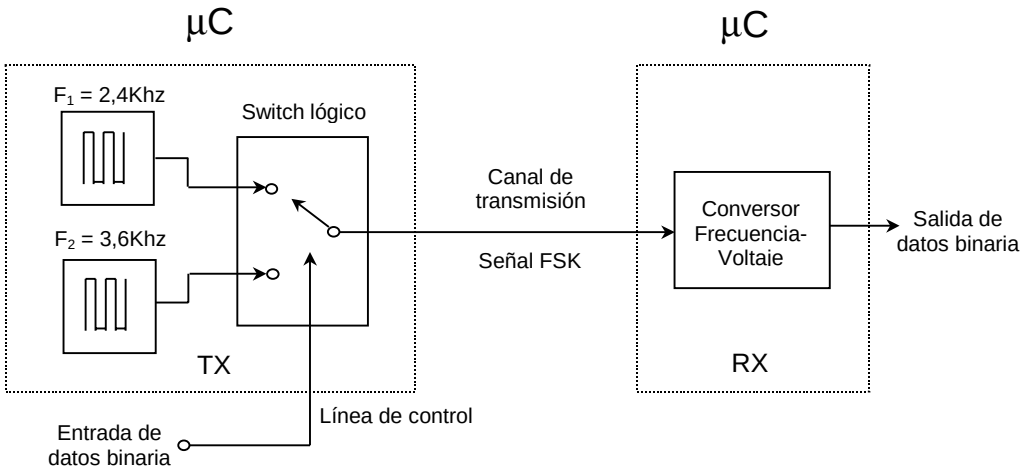


Figura 27 *Diagrama de bloques de los módulos de transmisión y recepción FSK*

donde se puede decir que el transmisor realiza un proceso que equivale a conmutar la línea de salida entre dos osciladores de diferente frecuencia, esta conmutación se hace de acuerdo a la entrada de datos binaria. En la etapa de recepción el conversor Frecuencia-Voltaje se encarga de recuperar los datos binarios y enviarlos a la salida.

8.8.1 Software del transmisor. El software del transmisor es el encargado de realizar la modulación FSK de la entrada digital. La información serial de la entrada digital, la cual es entregada por un Computador, es llevada al microcontrolador por medio de un driver de línea (MAX232), este se encarga de llevar la señal a niveles de voltaje TTL, correspondientes a los niveles de voltaje que maneja el microcontrolador.

Al dar inicio al programa principal, se setea el bit seis del puerto B (RB6), el cual entrega la señal de salida modulada, esto se hace para indicar que la salida se encuentra en estado de reposo. Luego el programa ingresa a un loop infinito, el cual solo es interrumpido por el desbordamiento del Timer 0 que genera una interrupción. Esta interrupción dirige el programa a la dirección de memoria 0004H donde se encuentra el vector de interrupciones. En este sitio se ubica un salto que apunta a la dirección donde se encuentra la rutina de servicio a la interrupción. Esta rutina es la que finalmente se encarga del procesamiento de la entrada digital (bit siete del puerto B).

La rutina de servicio a la interrupción es la encargada de realizar el muestreo de la entrada digital y de generar la señal de salida. Al iniciarse esta rutina se entra a la subrutina Tiempo de espera, la cual realiza un retardo de aproximadamente 150ms, la línea de entrada debe de estar en estado de reposo (uno lógico) durante el tiempo de espera para continuar con el programa, el caso contrario indicara que al momento de iniciarse el programa el Computador estaba realizando una transmisión y esta será descartada. Luego del tiempo de espera el programa queda listo para recibir el *Start bit* de una trama.

Al instante en que se detecta un *Start bit* se empieza a contar veinticuatro interrupciones (duración de un bit), durante las cuales se lee el estado lógico de la entrada, al completarse estas veinticuatro interrupciones se entra a la subrutina Txout donde se verifica el estado lógico del bit leído y se decide la frecuencia que deberá tener el bit modulado. Este proceso de conteo de veinticuatro interrupciones se repite con los ocho bits de datos y el *Stop bit*. Posterior a la verificación hecha por la subrutina Txout y de acuerdo a la frecuencia que deberá tener la señal de salida, se realiza un conteo de interrupciones durante los cuales se harán cambios de estado para formar la señal de onda cuadrada con la frecuencia necesaria.

Al terminar el muestreo y la modulación de la trama completa se verifica que el ultimo bit leído sea un *Stop bit* (uno lógico), si es así se realiza un reinicio del programa, es decir, se vuelve a las condiciones de espera de un próximo *Start bit*, pero en el caso en que el ultimo bit leído sea un cero lógico, significara que

hay un error de transmisión, por lo cual se dará un aviso de error y se entrara de nuevo a la subrutina Tiempo de espera.

A continuación se hace una descripción detallada de las rutinas implementadas para el desarrollo del modulador FSK.

8.8.1.1 Programa principal. En esta parte del programa se configura el Timer 0, el cual utiliza la fuente de reloj interna del microcontrolador, se programan el bit siete del puerto B como entrada y los bits cero a seis del puerto B como salida, se inicializan todos los registros que se utilizan en el programa, se habilita la interrupción del Timer 0 y se setea el bit seis del puerto B para indicar que la salida se encuentra en estado de reposo. Esto es realizado únicamente al inicio del programa y no se vuelve a entrar en el mientras el programa este corriendo. Luego el programa ingresa a un loop infinito, es decir, se ejecuta un salto que apunta a la dirección de memoria donde se encuentra el mismo salto, este lazo cerrado que se produce solo se interrumpe por la acción de desbordamiento del Timer 0 cada 34,6µs para procesar la entrada digital.

8.8.1.2 Rutina de servicio a la interrupción del Timer 0. Esta rutina inicialmente ejecuta un PUSH, el cual se encarga de almacenar el registro de trabajo W y las banderas de estado de la ALU como Carry, Digit Carry y Cero que se encuentran en el registro STATUS. Luego se realiza el ajuste de tiempo

del Timer 0, este ajuste se realiza para cargar el Timer 0 correctamente y de esta manera calibrar el tiempo en que se presentara la próxima interrupción.

A continuación se realiza una actualización del puerto B que consiste en mandar a las salidas del puerto los cambios que se hayan hecho al registro BUFFER. Después se verifica que la bandera TIEMPO sea uno, esto indica que el tiempo de espera ya ha sido ejecutado, en el caso en que sea cero se salta a la subrutina Tiempo de espera. El registro REGCON0 es el registro encargado de controlar la sincronización del programa con el *Start bit* de una trama, si este registro es uno indicara que ya se presento un *Start bit* por lo tanto se debe empezar el conteo de las veinticuatro interrupciones, este conteo lo realiza el registro REG1.

Al completarse veinticuatro interrupciones se llama a la subrutina Txout, esta se encarga de verificar el estado lógico del bit leído y decidir la frecuencia que deberá tener la señal modulada, esto se logra activando la bandera FREC que se encarga de indicar al programa la frecuencia que debe tener la señal de onda cuadrada que se utiliza como señal de salida.

El registro REG2 es quien lleva el conteo del numero de bits leídos, si este es diferente de cero indicara que ya se ha leído el primer bit, por lo tanto se debe empezar la modulación, para realizarla se verifica la bandera FREC, si esta es uno indicara que la portadora debe tener la frecuencia que representa un uno lógico que es de 3,6KHz, pero si FREC es cero la frecuencia de la portadora deberá ser de 2,4KHz que representa un cero lógico. Para formar la onda

cuadrada de 3,6KHz se hacen seis cambios de estado durante veinticuatro interrupciones y para formar la onda de 2,4KHz se hacen cuatro cambios de estado durante las veinticuatro interrupciones. Cada cambio de estado se envía al bit seis del registro BUFFER.

Al realizar la lectura del bit siete (RD) del puerto B, se verifica el estado lógico de la entrada digital, si es un cero lógico se llama a la subrutina Control, la cual se encarga de llevar el conteo del numero de ceros lógicos que se han leído y de controlar la sincronización del programa con el *Start bit*.

Si el registro REG2 es igual a diez quiere decir que ya se ha leído la trama completa, por lo que se llama a la subrutina Stop, que se encarga de verificar que el ultimo bit leído sea un *Stop bit* valido (uno lógico). Al haberse completado la lectura de la trama completa se produce un reinicio del programa, es decir, se prepara el programa para detectar un próximo *Start bit*, pero este reinicio se ejecuta veinte interrupciones después de que se ha leído el *Stop bit*, debido a que se debe permitir que este ultimo termine de ser modulado, ya que la modulación se efectúa después de realizar la lectura.

Por ultimo se ejecuta un POP para restaurar el registro W y las banderas de Carry, Digit Carry y Cero que se habían almacenado a su registro original (STATUS).

8.8.1.3 Subrutina Tiempo de espera. La subrutina de tiempo de espera consiste en un retardo de 150ms, esto se logra contando las interrupciones del Timer 0 que se producen cada 34,6μs, de esta manera se realiza el conteo de 256 interrupciones lo que proporciona un tiempo de 8,8ms, para lograr un tiempo de 150ms se cuentan 17 ciclos de 8,8ms.

Mientras se ejecuta el tiempo de espera se realiza una lectura de la entrada digital, si el resultado de la lectura es un uno lógico (estado de reposo) se sigue con el conteo hasta completar el tiempo de espera requerido, después de cumplido esto el registro TIEMPO, el cual actúa como bandera para informar que el tiempo de espera ha sido ejecutado, se carga con un uno y la bandera de ERROR (bit cinco del puerto B) se clarea, pero si la lectura da como resultado un cero lógico indicara que una transmisión ha comenzado antes de dar inicio al programa y esto provocara un error de transmisión, lo cual hace que se reinicie de nuevo el tiempo de espera y se setea la bandera de ERROR.

8.8.1.4 Subrutina Txout. Esta subrutina revisa el registro REG0, el cual contiene el numero de lecturas que correspondieron a un cero lógico durante las veinticuatro interrupciones, si de las veinticuatro lecturas realizadas trece o más fueron un cero lógico indicara que el bit leído es un cero, pero si de las veinticuatro lecturas tomadas doce o menos fueron un cero lógico quiere decir que el bit leído es un uno. De esta manera se realiza un promedio de las lecturas para descartar los posibles cambios en la señal debidos al ruido que se pueda presentar.

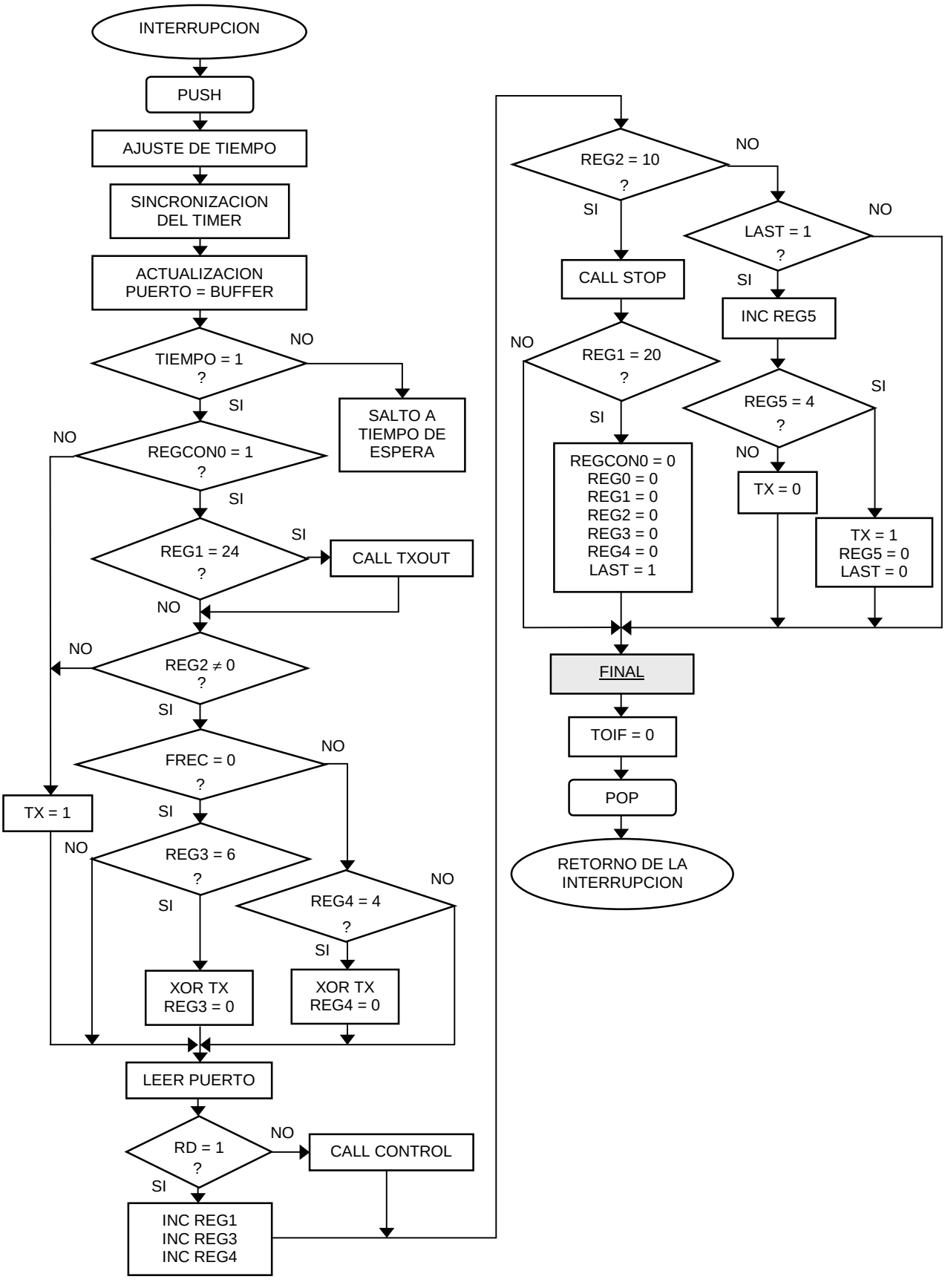
Si el bit leído es un cero, la señal de salida deberá tener una frecuencia de 2,4KHz que representa un cero lógico, de esta manera el registro FREC que indica que frecuencia se debe enviar a la salida se clarea, pero si el bit leído es un uno, el registro FREC se carga con un uno indicando que la señal de salida tendrá una frecuencia de 3,6KHz representando así un uno lógico. El estado inicial de la onda cuadrada en ambas frecuencias debe ser uno, este estado es puesto en el bit seis (TX) del registro BUFFER. El resultado de las veinticuatro lecturas, es decir, uno o cero lógico es almacenado en el registro REGAUX. Luego se prepara al programa para leer el próximo bit.

8.8.1.5 Subrutina Control. Esta subrutina es quien realiza el conteo de las lecturas que corresponden a un cero lógico, incrementando el registro REG0. También realiza la sincronización del programa con el *Start bit* de una trama, esta sincronización consiste en preparar al programa para empezar la lectura de todos los bits de la trama, esto se logra inicializando los registros encargados de este proceso. La subrutina Control utiliza el registro REGCON0 como bandera de control para indicar que la sincronización ya se efectuó, la sincronización solo se hace cuando se detecta el *Start bit* y no se realiza mientras se lee la trama completa, después de que esta ha sido leída y codificada el programa queda a la espera de un nuevo *Start bit* y de una nueva sincronización.

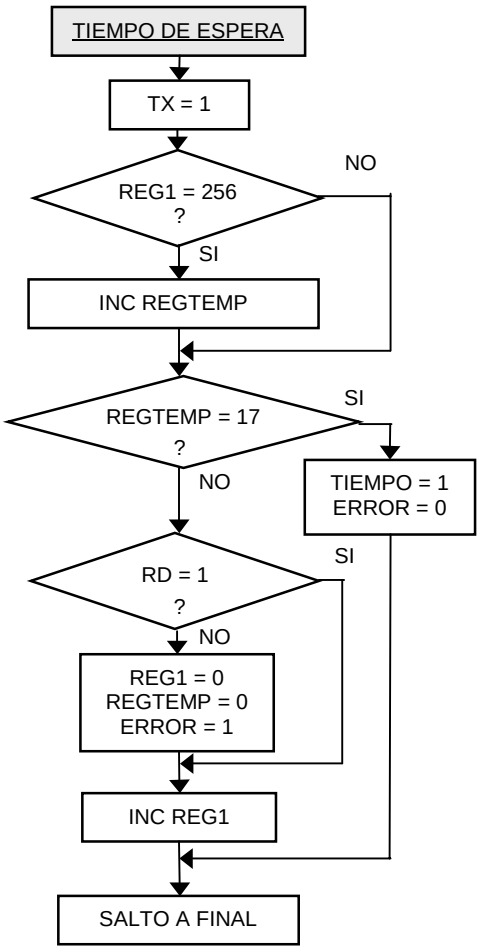
8.8.1.6 Subrutina Stop. Esta subrutina consiste en verificar si el ultimo bit leído (décimo bit) corresponde a un *Stop bit* valido, es decir, un uno lógico. La verificación se realiza revisando el contenido del registro REGAUX, el cual almacena el ultimo bit leído. Si el registro REGAUX es un uno, indicara que se leyó un *Stop bit* que es valido, de esta manera se clarea la bandera de ERROR y se retorna, pero en el caso contrario, es decir, si el registro REGAUX es cero indica que el *Stop bit* leído no es valido causando un error de comunicación, para lo cual se setea la bandera de ERROR y se clarea la bandera de TIEMPO obligando al programa a entrar de nuevo a la subrutina Tiempo de espera.

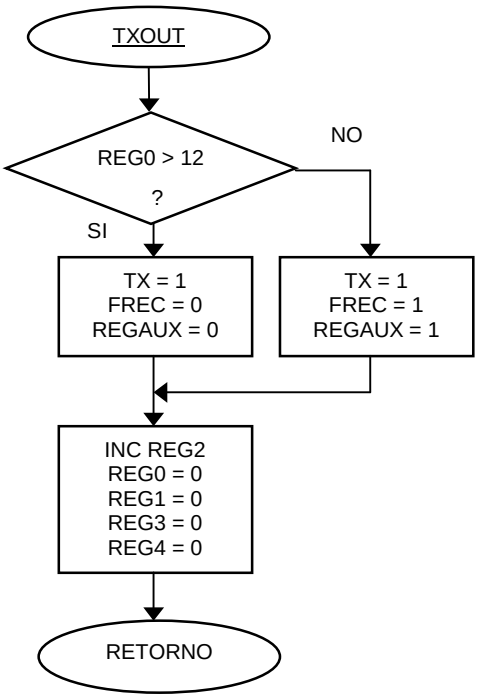
8.8.2 Diagrama de flujo del transmisor. A continuación se presenta el diagrama de flujo de la rutina de servicio a la interrupción y de las subrutinas que fueron necesarias implementar en el desarrollo de la etapa de transmisión para el módulo FSK.

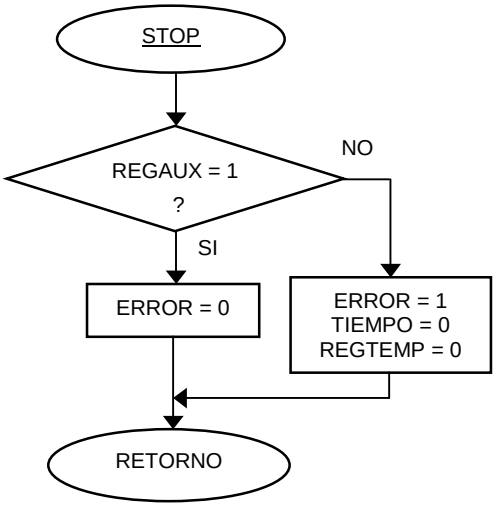
RUTINA DE SERVICIO A LA INTERRUPCION DEL TRANSMISOR FSK

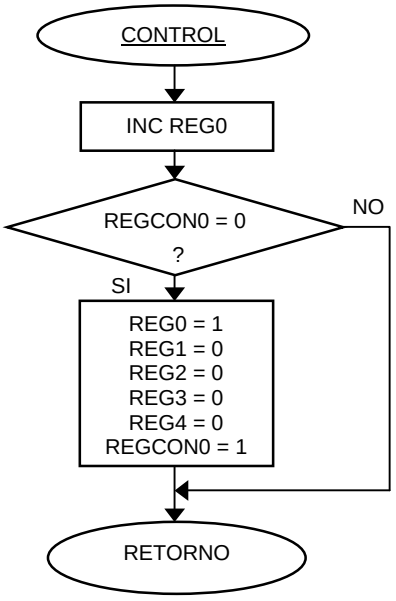


SUBRUTINAS


TIEMPO DE ESPERA
TX = 1
REG1 = 256 ?
NO
SI
INC REGTEMP
REGTEMP = 17 ?
SI
NO
TIEMPO = 1
ERROR = 0
RD = 1 ?
SI
NO
REG1 = 0
REGTEMP = 0
ERROR = 1
INC REG1
SALTO A FINAL


TXOUT
REG0 > 12 ?
NO
SI
TX = 1
FREC = 0
REGAUX = 0
TX = 1
FREC = 1
REGAUX = 1
INC REG2
REG0 = 0
REG1 = 0
REG3 = 0
REG4 = 0
RETORNO


STOP
REGAUX = 1 ?
NO
SI
ERROR = 0
ERROR = 1
TIEMPO = 0
REGTEMP = 0
RETORNO


CONTROL
INC REG0
REGCON0 = 0 ?
NO
SI
REG0 = 1
REG1 = 0
REG2 = 0
REG3 = 0
REG4 = 0
REGCON0 = 1
RETORNO

8.8.3 Programa en código fuente del transmisor. A continuación se da el programa detallado, que se utilizo en el microcontrolador, con sus respectivos comentarios.

```
;-------------------------------------------------------------------------------------------------------------------
;                                        Modulador FSK (Transmisor)
;-------------------------------------------------------------------------------------------------------------------
;       FSKTX.ASM

        list p=16c622           ;Procesador
        include <p16c622.inc>

;       Al programar el micro tener en cuenta los fusibles de configuración
;       OSC = HS, WDT = OFF, PWRTE = OFF, CP = OFF, BODEN = OFF


;------------------------------------------------- Zona de Registros ---------------------------------------------

tmr_opt     equ     01h     ;Timer0 / option
status      equ     03h     ;
intcon      equ     0bh     ;
puertob     equ     06h     ;
buffer      equ     2ah     ;Registro que almacena cambios del puertob
reg0        equ     2bh     ;Contador de ceros
reg1        equ     2ch     ;Contador de interrupciones
reg2        equ     2dh     ;Contador de bits
reg3        equ     2eh     ;Contador de paquetes de 4 interrupciones
reg4        equ     2fh     ;Contador de paquetes de 6 interrupciones
reg5        equ     30h     ;Contador de las ultimas 4 interrupciones del stop bit
reg6        equ     31h     ;Contador de interrupciones para oscilador1
reg7        equ     32h     ;Contador de interrupciones para oscilador2
regaux      equ     33h     ;Registro auxiliar que almacena el contenido de tx
regcon0     equ     34h     ;Bandera de control de start-bit
tempwr      equ     35h     ;Temporal de reloj
tiempo      equ     36h     ;Bandera de control del tiempo de espera
regtemp     equ     37h     ;Contador para determinar el tiempo de espera
frec        equ     38h     ;Bandera de frecuencia de la portadora
last        equ     39h     ;Bandera para habilitar Tx en las 5 int finales
wtemp       equ     70h     ;Temporal de w
stemp       equ     71h     ;Temporal de status


;------------------------------------------------- Asignaciones de bit -------------------------------------------

rd          equ     7       ;Entrada serial (RS-232)
tx          equ     6       ;Salida serial (FSK)
err         equ     5       ;Aviso de error
gie         equ     7       ;Habilita int. general
toie        equ     5       ;Habilita int. por tmr0
toif        equ     2       ;Flag de int. por tmr0
rp0         equ     5       ;Página de memoria
osc1        equ     4       ;Portadora
osc2        equ     3       ;Portadora desplazada
sincro      equ     2       ;Sincronización del osciloscopio
```

```
;------------------------------------------------ Configuración del PIC ----------------------------------------------

        org     0
        goto    inicio          ;Inicio de programa principal

        org     4
        goto    inter           ;Vector de interrupciones

        org     5               ;Inicio programa "inicio"
inicio  bsf     status,rp0      ;Va al banco1
        movlw   08h             ;Option, clock interno
        movwf   tmr_opt         ;
        movlw   80h             ;Configuración de puertos
        movwf   puertob         ;Puertob 7:entrada, 6-0:salida
        bcf     status,rp0      ;Retorno banco0
        clrf    tmr_opt         ;Inicialización de registros
        clrf    puertob         ;
        clrf    reg0            ;
        clrf    reg1            ;
        clrf    reg2            ;
        clrf    reg3            ;
        clrf    reg4            ;
        clrf    reg5            ;
        clrf    reg6            ;
        clrf    reg7            ;
        clrf    regaux          ;
        clrf    regcon0         ;
        clrf    buffer          ;
        clrf    tempwr          ;
        clrf    tiempo          ;
        clrf    regtemp         ;
        clrf    frec            ;
        clrf    last            ;
        bsf     intcon,gie      ;Habilitación general de interrupciones
        bsf     intcon,toie     ;Habilita interrupción de timer0
        bsf     buffer,tx       ;Inicialización de salida serial tx
        bsf     buffer,osc1     ;Inicialización de señales generadas
        bsf     buffer,osc2     ;
        bcf     buffer,sincro   ;

;------------------------------------------------------------ Loop ------------------------------------------------------------

loop    goto    loop            ;Programa principal, loop infinito

;------------------------- RUTINA DE SERVICIO A LA INTERRUPCION DEL TMRO ----------------------

inter   movwf   wtemp           ;Push
        swapf   status,w        ;
        movwf   stemp           ;

        movf    tmr_opt,w       ;Ajuste de tiempo del timer0
        movwf   tempwr          ;
        btfss   tempwr,0        ;
        goto    time61          ;

time61  movlw   61h             ;61h -> 34,6 useg (aproximadamente)
        movwf   tmr_opt         ;Carga del timer0

princip movf    buffer,w        ;
        movwf   puertob         ;Actualización del puerto b
```

```
        btfss   tiempo,0        ;Se verifica si ya se ejecuto el tiempo de espera
        goto    time            ;

        btfss   regcon0,0       ;Mientras no se de un start bit, tx es 1 (reposo)
        goto    jump            ;

;-------------------------------------------------- Conversión ----------------------------------------------------

        movlw   18h             ;
        subwf   reg1,w          ;Reg1 = 24?
        btfsc   status,2        ;Si reg1 = 24 -> call txout
        call    txout           ;Actualización de tx

        movf    reg2,w          ;Si reg2 es diferente de cero se permite
        btfsc   status,2        ;la conversión del primer bit leído
        goto    jump            ;

cero    btfsc   frec,0          ;Si frec es cero se convierte un cero
        goto    uno             ;
        movlw   06h             ;Se cuentan 6 interrupciones
        subwf   reg3,w          ;
        btfss   status,2        ;
        goto    leer            ;

        movlw   40h             ;XOR de Tx cada 6 interrupciones
        xorwf   buffer,f        ;
        clrf    reg3            ;
        goto    leer            ;

uno     movlw   04h             ;Si frec es uno se convierte un uno
        subwf   reg4,w          ;Se cuentan 4 interrupciones
        btfss   status,2        ;
        goto    leer            ;

        movlw   40h             ;XOR de tx cada 4 interrupciones
        xorwf   buffer,f        ;
        clrf    reg4            ;
        goto    leer            ;

jump    bsf     buffer,tx       ;Estado de reposo de la línea

;-------------------------------------------------- Muestreo ----------------------------------------------------

leer    btfss   puertob,rd      ;Lectura de la entrada serial
        call    control         ;
        incf    reg1,f          ;Incrementa contador de interrupciones
        incf    reg3,f          ;Incrementa contador de paquetes de 4 interrupciones
        incf    reg4,f          ;Incrementa contador de paquetes de 6 interrupciones
        incf    reg6,f          ;Incrementa contador de oscilador1
        incf    reg7,f          ;Incrementa contador de oscilador2
```

```
;---------------------------------------------------------- Stop ----------------------------------------------------------

stopbit  movlw  0ah             ;Verificación del stop (bit 10)
         subwf  reg2,w          ;
         btfss  status,2        ;Reg2 = 10?
         goto   ultimo          ;
         bcf    buffer,err      ;Se asume error = 0
         btfsc  regaux,0        ;Chequeo del bit 10 para definir si hay error
         goto   valido          ;Tx = 1 para el bit 10, stop valido
         bsf    buffer,err      ;Tx = 0 para el bit 10, stop no valido
         clrf   tiempo          ;Se inicia de nuevo el tiempo de espera
         clrf   regtemp         ;
         goto   final           ;

;---------------------------------------------------------- Reset ----------------------------------------------------------

valido   movlw  14h             ;El Reset se da en la interrupción 20 después
         subwf  reg1,w          ;de que se ha leído el stop bit
         btfss  status,2        ;
         goto   final           ;
         clrf   regcon0         ;Reset
         clrf   reg0            ;
         clrf   reg1            ;
         clrf   reg2            ;
         clrf   reg3            ;
         clrf   reg4            ;
         bsf    last,0          ;Habilita la salida de tx en las 4 int. finales
         goto   final           ;

;---------------------------------------------------------- Last ----------------------------------------------------------

ultimo   btfss  last,0          ;
         goto   final           ;


         incf   reg5,f          ;Conteo de las 4 interrupciones faltantes
         movlw  04h             ;
         subwf  reg5,w          ;
         btfss  status,2        ;
         goto   recover         ;


         clrf   last            ;Finalización de la recuperación del 4 segmento
         clrf   reg5            ;
         movlw  04h             ;Final de la trama, se permite que se vuelva a
         xorwf  buffer,f        ;sincronizar con el osciloscopio (bit 2 del puertob)
         bsf    buffer,tx       ;
         goto   final           ;


recover  bcf    buffer,tx       ;Recuperación del cuarto segmento
         goto   final           ;
```

```
;---------------------------------------- Subrutina tiempo de espera ----------------------------------------

time    bsf     buffer,tx       ;Mientras se ejecuta el tiempo de espera, tx es 1 (reposo)
        movlw   0ffh            ;Tiempo de espera de aproximadamente 150mseg
        subwf   reg1,w          ;34.6useg * 256 = 8.8mseg
        btfsc   status,2        ;
        incf    regtemp,f       ;Se cuentan ciclos de 8.8mseg

        movlw   11h             ;
        subwf   regtemp,w       ;8.8mseg * 17 = 150.5mseg
        btfss   status,2        ;Regtemp = 17 -> tiempo = 1 y error = 0
        goto    again           ;
        bsf     tiempo,0        ;Cumplidos los 150mseg se setea la bandera de tiempo
        bcf     buffer,err      ;Se clarea la bandera de error
        goto    final           ;

again   btfsc   puertob,rd      ;Lectura de la entrada serial
        goto    unor            ;
        clrf    reg1            ;Si rd es cero durante el tiempo de espera
        clrf    regtemp         ;se inicia de nuevo el conteo
        bsf     buffer,err      ;Aviso de error por transmisión en proceso
unor    incf    reg1,f          ;Incremento del contador de interrupciones


;------------------------------------------------------------ Final ------------------------------------------------------------

final   movlw   06h             ;Generación de la portadora
        subwf   reg6,w          ;(bit 4 del puertob)
        btfss   status,2        ;
        goto    shift           ;
        movlw   10h             ;
        xorwf   buffer,f        ;
        clrf    reg6            ;
shift   movlw   04h             ;Generación de la portadora desplazada
        subwf   reg7,w          ;en frecuencia (bit 3 del puertob)
        btfss   status,2        ;
        goto    salir           ;
        movlw   08h             ;
        xorwf   buffer,f        ;
        clrf    reg7            ;

salir   bcf     intcon,toif     ;Se clarea la bandera del tmr0
        swapf   stemp,w         ;Pop
        movwf   status          ;
        swapf   wtemp,f         ;
        swapf   wtemp,w         ;
        retfie                  ;

;---------------------------------------------------- Subrutina txout ----------------------------------------------------

txout   movf    reg0,w          ;
        sublw   0ah             ;(10 - reg0)
        btfss   status,0        ;Si reg0 > 10 -> Tx=0
        goto    tx0             ;
tx1     bsf     buffer,tx       ;Tx=1
        bsf     frec,0          ;
        bsf     regaux,0        ;Se almacena bit leído en Regaux
        goto    done            ;
```

```
tx0     bsf     buffer,tx       ;Tx=0
        bcf     frec,0          ;
        bcf     regaux,0        ;Se almacena bit leído en Regaux

done    incf    reg2,f          ;
        clrf    reg0            ;Iniciación de bit
        clrf    reg1            ;
        clrf    reg3            ;
        clrf    reg4            ;
        return                  ;

;-------------------------------------------------- Subrutina control ----------------------------------------------------

control incf    reg0,f          ;
        movf    regcon0,w       ;Chequea si regcon0 = 0
        btfsc   status,2        ;
        call    sinc            ;Regcon0 = 0 -> Sincronización con Start bit
        return                  ;

;-------------------------------------------------- Sincronización -------------------------------------------------------

sinc    movlw   04h             ;Señal de sincronización del osciloscopio
        xorwf   buffer,f        ;con el inicio de la trama (bit 2 del puertob)
        clrf    reg0            ;Sincronización con el start bit
        clrf    reg1            ;
        clrf    reg2            ;
        clrf    reg3            ;
        clrf    reg4            ;
        incf    reg0,f          ;
        bsf     regcon0,0       ;Se deshabilitan futuras sincronizaciones
        return                  ;por un 0. Se restablece con el Stop bit

;-------------------------------------------------------------------------------------------------------------------------
        end
```

8.8.4 Software del receptor. El software del receptor es el encargado de demodular la señal FSK y recuperar la información digital. Este software divide el bit modulado en veinticuatro segmentos, cada segmento esta compuesto por una interrupción, durante la cual se realiza la lectura de la entrada modulada, al final de la interrupción se define el estado lógico del segmento leído y se almacena en el registro STORE1, STORE2 o STORE3, la razón de usar tres registros para almacenar es que cada registro es de ocho bits y se necesitan tres para almacenar veinticuatro segmentos. Al terminar de leer el bit modulado, en los registros STORE1, STORE2 y STORE3 quedara guardada la secuencia de segmentos y esta ayudara a determinar si la frecuencia del bit modulado corresponde a 2,4KHz o a 3,6KHz.

Al inicio del programa principal se setea el bit seis del puerto B (RB6), el cual entrega la señal de salida demodulada, esto se hace para indicar que la salida se encuentra en estado de reposo. Luego el programa ingresa a un loop infinito, el cual solo es interrumpido por el desbordamiento del Timer 0 que genera una interrupción. Esta interrupción dirige el programa a la dirección de memoria 0004H donde se encuentra el vector de interrupciones. En este sitio se ubica un salto que apunta a la dirección donde se encuentra la rutina de servicio a la interrupción. Es esta rutina la que finalmente se encarga del procesamiento de la entrada modulada (bit siete del puerto B).

La rutina de servicio a la interrupción es la encargada de realizar el muestreo de la entrada modulada y de generar la señal de salida. Al iniciarse esta rutina se entra a la subrutina Tiempo de espera, la cual realiza un retardo de

aproximadamente 150ms en el cual se verifica que el Computador no este transmitiendo al momento de iniciar el programa.

Luego de ejecutar el tiempo de espera el programa queda listo para recibir el *Start bit* modulado de una trama. Después de detectar un *Start bit* se entra a la subrutina Chequeo donde se verifica el estado lógico del segmento leído y se almacena, este proceso es repetido hasta obtener los veinticuatro segmentos correspondientes a un bit modulado, lo cual toma veinticuatro interrupciones, al final de las cuales se determina la frecuencia del bit modulado y se define el estado lógico de la salida digital. Los ocho bits de datos y el *Stop bit* son demodulados de la misma forma.

Al terminar el muestreo y la demodulación de la trama completa, se verifica que el ultimo bit leído sea un *Stop bit* valido, si es así se realiza un reinicio del programa, es decir, se vuelve a las condiciones de espera de un próximo *Start bit*, pero en el caso en que el ultimo bit leído sea un cero lógico, significara que hay un error de transmisión, por lo cual se dará un aviso de error y se entrara de nuevo a la subrutina Tiempo de espera.

A continuación se hace una descripción detallada de las rutinas implementadas para el desarrollo del demodulador FSK.

8.8.4.1 Programa principal. En esta parte del programa se configura el Timer 0, el cual utiliza la fuente de reloj interna del microcontrolador, se programan el

bit siete del puerto B como entrada y los bits cero a seis del puerto B como salida, se inicializan todos los registros que se utilizan en el programa, se habilita la interrupción del Timer 0 y se setea el bit seis del puerto B para indicar que la salida se encuentra en estado de reposo. Esto es realizado únicamente al inicio del programa y no se vuelve a entrar en el mientras el programa este corriendo. Luego el programa ingresa a un loop infinito, es decir, se ejecuta un salto que apunta a la dirección de memoria donde se encuentra el mismo salto, este lazo cerrado que se produce solo se interrumpe por la acción de desbordamiento del Timer 0 cada 34,6μs para procesar la entrada modulada.

8.8.4.2 Rutina de servicio a la interrupción del Timer 0. Esta rutina inicialmente ejecuta un PUSH, el cual se encarga de almacenar el registro de trabajo W y las banderas de estado de la ALU. Luego se realiza el ajuste de tiempo del Timer 0, este ajuste se realiza para cargar el Timer 0 correctamente y de esta manera calibrar el tiempo en que se presentara la próxima interrupción.

Posteriormente se realiza una actualización del puerto B que consiste en mandar a las salidas del puerto los cambios que se hayan hecho al registro BUFFER. Después se verifica que la bandera TIEMPO sea uno, esto indica que el tiempo de espera ya ha sido ejecutado, en el caso en que sea cero se salta a la subrutina Tiempo de espera. El registro REGCON0 es el registro encargado de controlar la sincronización del programa con el *Start bit*

modulado, si este registro es uno indicara que ya se presento un *Start bit* por lo tanto se debe empezar el conteo de interrupciones, este conteo lo realiza el registro REG1.

Durante cada interrupción se llama a la subrutina Chequeo, esta se encarga de verificar y almacenar el estado lógico del segmento leído, esto se repite hasta completar veinticuatro segmentos, lo cual toma el mismo numero de interrupciones, al completar veinticuatro segmentos se determina la frecuencia del bit modulado y se define el estado lógico de la salida digital. Este proceso se repite con los ocho bits de datos y el *Stop bit*.

Al realizar la lectura del bit siete (RD) del puerto B, se verifica el estado lógico de la entrada modulada, si es un cero lógico se llama a la subrutina Control, la cual se encarga de llevar el conteo del numero de ceros lógicos que se han leído y de controlar la sincronización del programa con el *Start bit* modulado.

Si el registro REG2 es igual a diez, indicara que se ha leído la trama completa, por lo que se llama a la subrutina Stop, que se encarga de verificar que el ultimo bit leído sea un *Stop bit* valido. Dentro de esta subrutina se produce el reinicio del programa, es decir, se prepara al programa para detectar un próximo *Start bit*.

Por ultimo se ejecuta un POP para restaurar el registro W y las banderas de estado de la ALU a su registro original (STATUS).

8.8.4.3 Subrutina Tiempo de espera. La subrutina de tiempo de espera consiste en un retardo de 150ms, esto se logra contando las interrupciones del Timer 0 que se producen cada 34,6μs, de esta manera se realiza el conteo de 256 interrupciones lo que proporciona un tiempo de 8,8ms, para lograr un tiempo de 150ms se cuentan 17 ciclos de 8,8ms.

Mientras se ejecuta el tiempo de espera se realiza una lectura de la entrada modulada, si el resultado de la lectura es un uno lógico (estado de reposo) se sigue con el conteo hasta completar el tiempo de espera requerido, después de cumplido esto el registro TIEMPO, el cual actúa como bandera para informar que el tiempo de espera ha sido ejecutado, se carga con un uno y la bandera de ERROR (bit cinco del puerto B) se clarea, pero si la lectura da como resultado un cero lógico indicara que una transmisión ha comenzado antes de dar inicio al programa y esto provocara un error de transmisión, lo cual hace que se reinicie de nuevo el tiempo de espera y se setea la bandera de ERROR.

8.8.4.4 Subrutina Chequeo. Esta subrutina revisa el registro REG0 durante cada interrupción, el REG0 contiene la lectura correspondiente a un cero lógico durante la interrupción, si el registro REG0 es igual a uno esto indicara que el segmento leído es un cero, lo contrario quiere decir que el segmento leído es un uno. De esta manera se acumulan veinticuatro segmentos durante veinticuatro interrupciones, cada uno de los segmentos es almacenado, los primeros ocho segmentos leídos son almacenados en el registro STORE1, los

ocho segmentos siguientes son almacenados en el registro STORE2 y los últimos ocho segmentos son almacenados en el registro STORE3.

Al completar los veinticuatro segmentos en los registros STORE1, STORE2 y STORE3 queda una secuencia de unos y ceros que ayuda a determinar la frecuencia del bit modulado. Se toma en cuenta el desplazamiento de las secuencias ya que se puede presentar que las lecturas queden desplazadas un segmento a la izquierda o a la derecha, de esta manera si el registro STORE1 tiene una secuencia igual a 1EH, si el registro STORE2 tiene la secuencia E0H y si el registro STORE3 tiene una secuencia igual a 01H indicaran que la frecuencia del bit modulado es de 2,4KHz, por lo tanto la salida digital, por la cual se obtiene el bit demodulado, deberá tener un estado lógico cero, pero si los registros STORE1, STORE2 y STORE3 tienen secuencias iguales a 06H indicaran que la frecuencia del bit modulado es de 3,6KHz y la salida digital deberá tener un estado lógico uno. Luego de esto se prepara al programa para leer el próximo bit.

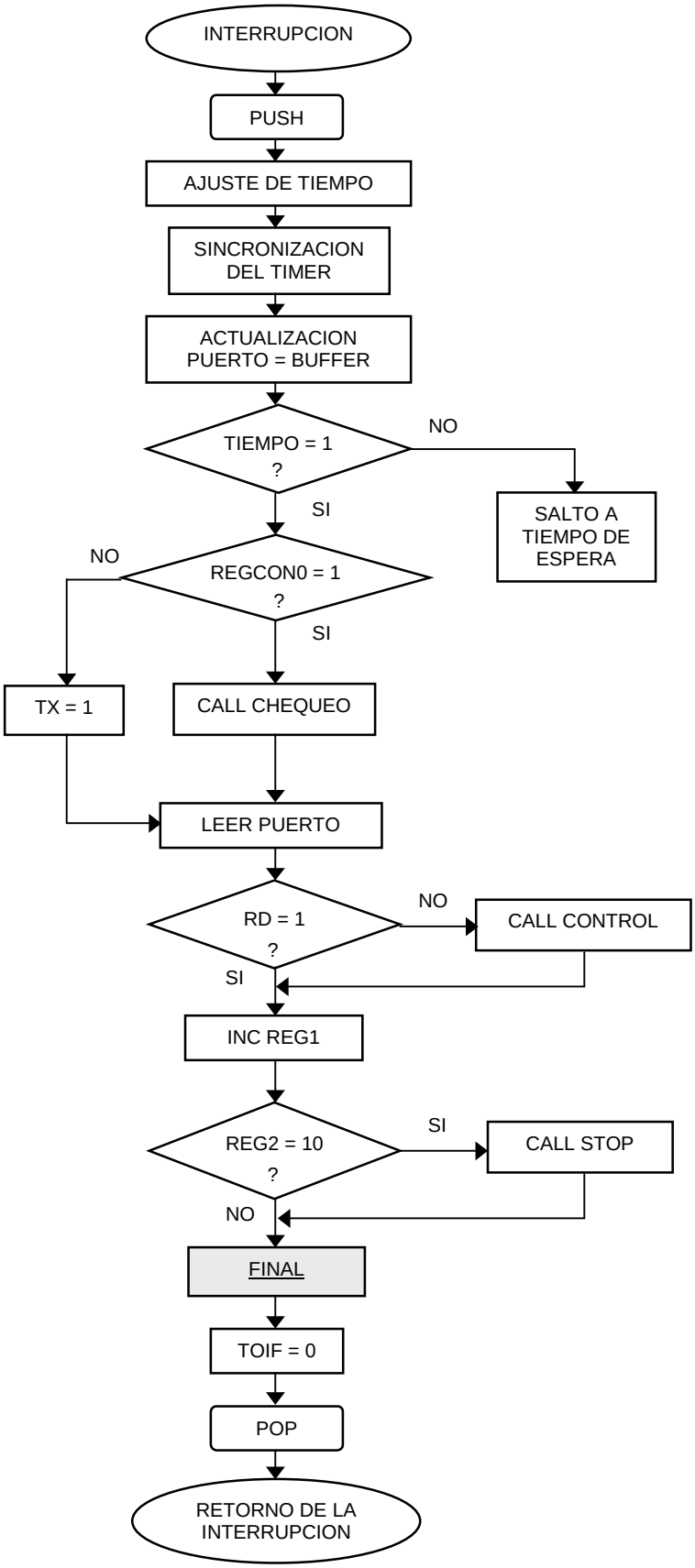
8.8.4.5 Subrutina Control. Esta subrutina es quien realiza el conteo de las lecturas que corresponden a un cero lógico, incrementando el registro REG0. También realiza la sincronización del programa con el *Start bit* modulado de una trama, esta sincronización consiste en preparar al programa para empezar la lectura de todos los bits modulados, esto se logra inicializando los registros encargados de este proceso. La subrutina Control utiliza el registro REGCON0 como bandera de control para indicar que la sincronización ya se efectuó, la

sincronización solo se hace cuando se detecta el *Start bit* modulado y no se realiza mientras se lee la trama modulada, después de que esta ha sido leída y demodulada el programa queda a la espera de un nuevo *Start bit* y de una nueva sincronización.

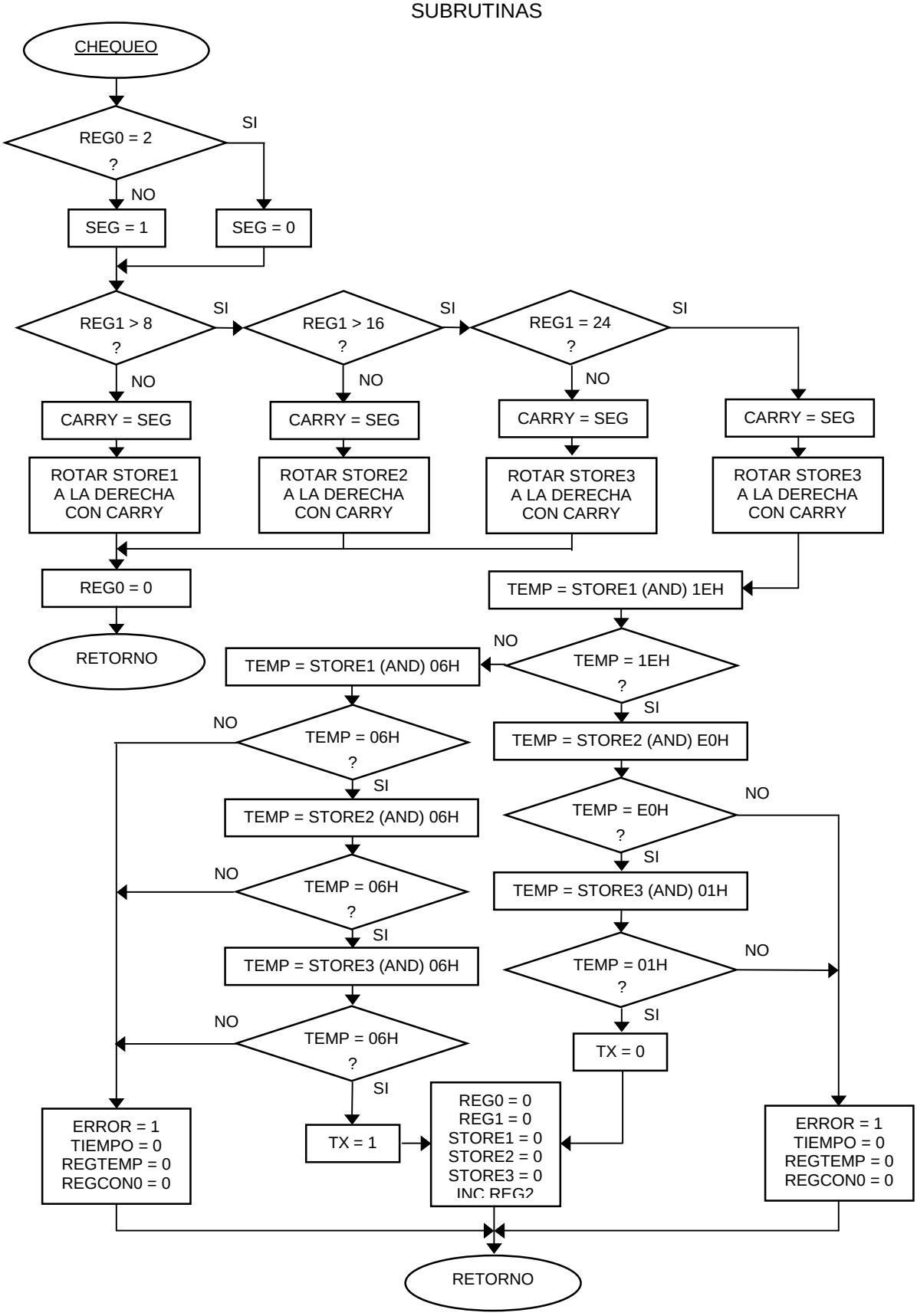
8.8.4.6 Subrutina Stop. Esta subrutina consiste en verificar que el ultimo bit leído (décimo bit) corresponda a un *Stop bit* valido, es decir, un uno lógico. La verificación se realiza revisando el contenido del bit seis (TX) del registro BUFFER, el cual contiene el estado lógico del ultimo bit demodulado. Si el estado de TX es uno indicara que se presento un *Stop bit* valido, pero si TX es cero se produce un error de comunicación, para lo cual se setea la bandera de ERROR y se clarea la bandera de TIEMPO obligando al programa a entrar de nuevo a la subrutina Tiempo de espera. En el caso de que el *Stop bit* sea valido se clarea la bandera de ERROR y se realiza el reinicio del programa, es decir, se prepara al programa para recibir un nuevo *Start bit* modulado.

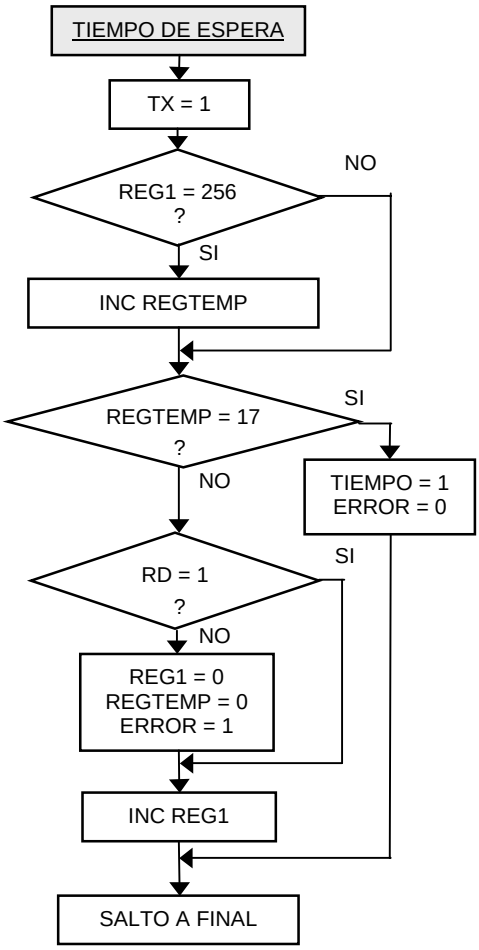
8.8.5 Diagrama de flujo del receptor. A continuación se presenta el diagrama de flujo de la rutina de servicio a la interrupción y de las subrutinas que fueron necesarias implementar en el desarrollo de la etapa de recepción para el módulo FSK.

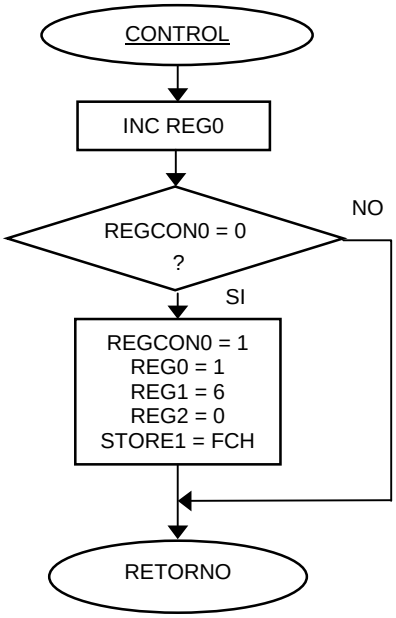
RUTINA DE SERVICIO A LA INTERRUPCION DEL RECEPTOR FSK

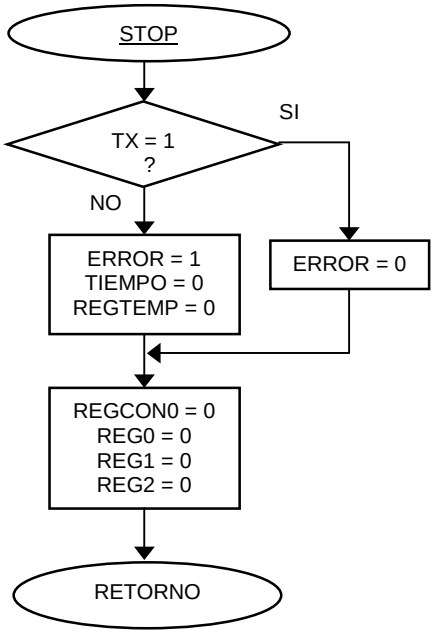


SUBRUTINAS


CHEQUEO
REG0 = 2 ?
SI
NO
SEG = 1
SEG = 0
REG1 > 8 ?
SI
REG1 > 16 ?
SI
REG1 = 24 ?
SI
NO
NO
NO
CARRY = SEG
CARRY = SEG
CARRY = SEG
CARRY = SEG
ROTAR STORE1 A LA DERECHA CON CARRY
ROTAR STORE2 A LA DERECHA CON CARRY
ROTAR STORE3 A LA DERECHA CON CARRY
ROTAR STORE3 A LA DERECHA CON CARRY
REG0 = 0
TEMP = STORE1 (AND) 1EH
RETORNO
NO
TEMP = 1EH ?
TEMP = STORE1 (AND) 06H
SI
NO
TEMP = 06H ?
TEMP = STORE2 (AND) E0H
SI
NO
TEMP = E0H ?
TEMP = STORE2 (AND) 06H
SI
NO
TEMP = 06H ?
TEMP = STORE3 (AND) 01H
SI
NO
TEMP = 01H ?
TEMP = STORE3 (AND) 06H
SI
NO
TEMP = 06H ?
TX = 0
SI
ERROR = 1
TIEMPO = 0
REGTEMP = 0
REGCON0 = 0
TX = 1
REG0 = 0
REG1 = 0
STORE1 = 0
STORE2 = 0
STORE3 = 0
INC REG2
ERROR = 1
TIEMPO = 0
REGTEMP = 0
REGCON0 = 0
RETORNO


TIEMPO DE ESPERA
TX = 1
REG1 = 256 ?
NO
SI
INC REGTEMP
REGTEMP = 17 ?
SI
NO
TIEMPO = 1
ERROR = 0
RD = 1 ?
SI
NO
REG1 = 0
REGTEMP = 0
ERROR = 1
INC REG1
SALTO A FINAL


CONTROL
INC REG0
REGCON0 = 0 ?
NO
SI
REGCON0 = 1
REG0 = 1
REG1 = 6
REG2 = 0
STORE1 = FCH
RETORNO


STOP
TX = 1 ?
SI
NO
ERROR = 1
TIEMPO = 0
REGTEMP = 0
ERROR = 0
REGCON0 = 0
REG0 = 0
REG1 = 0
REG2 = 0
RETORNO

8.8.6 Programa en código fuente del receptor. A continuación se da el programa detallado, que se utilizo en el microcontrolador, con sus respectivos comentarios.

```
;-------------------------------------------------------------------------------------------------------------------
;                                        Demodulador FSK (Receptor)
;-------------------------------------------------------------------------------------------------------------------
;       FSKRX.ASM

        list p=16c622           ;Procesador
        include <p16c622.inc>

;       Al programar el micro tener en cuenta los fusibles de configuración
;       OSC = HS, WDT = OFF, PWRTE = OFF, CP = OFF, BODEN = OFF


;------------------------------------------------- Zona de Registros ---------------------------------------------


tmr_opt     equ     01h     ;Timer0 / option
status      equ     03h     ;
intcon      equ     0bh     ;
puertob     equ     06h     ;
buffer      equ     2ah     ;Registro que almacena cambios del puertob
reg0        equ     2bh     ;Contador de ceros
reg1        equ     2ch     ;Contador de interrupciones
reg2        equ     2dh     ;Contador de bits
regcon0     equ     2eh     ;Bandera de control de start bit
tempwr      equ     2fh     ;Temporal del reloj
store1      equ     30h     ;Registro que almacena ocho segmentos leídos
store2      equ     31h     ;Registro que almacena ocho segmentos leídos
store3      equ     32h     ;Registro que almacena ocho segmentos leídos
temp        equ     33h     ;Registro temporal
tiempo      equ     34h     ;Bandera de control del tiempo de espera
regtemp     equ     35h     ;Contador para determinar el tiempo de espera
seg         equ     36h     ;Registro que guarda el segmento actual
wtemp       equ     70h     ;Temporal de w
stemp       equ     71h     ;Temporal de status


;------------------------------------------------- Asignaciones de bit -------------------------------------------


rd          equ     7       ;Entrada serial (FSK)
tx          equ     6       ;Salida serial (RS-232)
err         equ     5       ;Aviso de error
gie         equ     7       ;Habilita int. general
toie        equ     5       ;Habilita int. por tmr0
toif        equ     2       ;Flag de int. por tmr0
rp0         equ     5       ;Página de memoria
sincro      equ     2       ;Sincronización del osciloscopio
```

```
;------------------------------------------------ Configuración del PIC ----------------------------------------------

        org     0
        goto    inicio          ;Inicio de programa principal

        org     4
        goto    inter           ;Vector de interrupción

        org     5               ;Inicio programa "inicio"

inicio  bsf     status,rp0      ;Va al banco1
        movlw   08h             ;Option, clock interno
        movwf   tmr_opt         ;
        movlw   80h             ;Configuración de puertos
        movwf   puertob         ;Puertob 7:entrada, 6-0:salida
        bcf     status,rp0      ;Retorno al banco0

        clrf    tmr_opt         ;Inicialización de registros
        clrf    puertob         ;
        clrf    reg0            ;
        clrf    reg1            ;
        clrf    reg2            ;
        clrf    regcon0         ;
        clrf    buffer          ;
        clrf    tempwr          ;
        clrf    store1          ;
        clrf    store2          ;
        clrf    store3          ;
        clrf    temp            ;
        clrf    tiempo          ;
        clrf    regtemp         ;
        clrf    seg             ;

        bsf     intcon,gie      ;Habilitación general de interrupciones
        bsf     intcon,toie     ;Habilita interrupción del timer0
        bsf     buffer,tx       ;Inicialización del puerto b
        bcf     buffer,sincro   ;

;------------------------------------------------------------ Loop ------------------------------------------------------------

loop    goto    loop            ;Programa principal, loop infinito


;------------------------- RUTINA DE SERVICIO A LA INTERRUPCION DEL TMRO ----------------------


inter   movwf   wtemp           ;Push
        swapf   status,w        ;
        movwf   stemp           ;

        movf    tmr_opt,w       ;Ajuste de tiempo del timer0
        movwf   tempwr          ;
        btfss   tempwr,0        ;
        goto    time61          ;

time61  movlw   61h             ;61h -> 34,6 useg (aproximadamente)
        movwf   tmr_opt         ;Carga del timer0

princip movf    buffer,w        ;
        movwf   puertob         ;Actualización del puerto b
```

```
        btfss   tiempo,0        ;Se verifica si ya se ejecuto el tiempo de espera
        goto    time            ;

        btfss   regcon0,0       ;Mientras no se de un start bit, tx es 1 (reposo)
        goto    jump            ;

;-------------------------------------------------- Conversión ----------------------------------------------------

        call    check           ;Chequeo de la muestra obtenida
        goto    leer            ;

jump    bsf     buffer,tx       ;Estado de reposo de la línea

;-------------------------------------------------- Muestreo ------------------------------------------------------

leer    btfss   puertob,rd      ;Lectura de la entrada serial
        call    control         ;
        incf    reg1,f          ;Incrementa contador de interrupciones

stopbit movlw   0ah             ;Verificación del stop (bit 10)
        subwf   reg2,w          ;
        btfsc   status,2        ;Reg2 = 10?
        call    stop            ;Chequea si el bit 10 fue un 1
        goto    final           ;

;-------------------------------------------- Subrutina tiempo de espera ------------------------------------------

time    bsf     buffer,tx       ;Mientras se ejecuta el tiempo de espera, tx es 1 (reposo)
        movlw   0ffh            ;Tiempo de espera de aproximadamente 150mseg
        subwf   reg1,w          ;34.6useg * 256 = 8.8mseg
        btfsc   status,2        ;
        incf    regtemp,f       ;Se cuentan ciclos de 8.8mseg

        movlw   11h             ;
        subwf   regtemp,w       ;8.8mseg * 17 = 150.5mseg
        btfss   status,2        ;Regtemp = 17 -> tiempo = 1 y error = 0
        goto    again           ;
        bsf     tiempo,0        ;Cumplidos los 150mseg se setea la bandera de tiempo
        bcf     buffer,err      ;Se clarea la bandera de error
        goto    final           ;
again   btfsc   puertob,rd      ;Lectura de la entrada serial
        goto    unor            ;
        clrf    reg1            ;Si rd es cero durante el tiempo de espera
        clrf    regtemp         ;se inicia de nuevo el conteo
        bsf     buffer,err      ;
unor    incf    reg1,f          ;Incremento del contador de interrupciones

;------------------------------------------------------------ Final ---------------------------------------------------

final   bcf     intcon,toif     ;Se clarea la bandera del tmr0
        swapf   stemp,w         ;Pop
        movwf   status          ;
        swapf   wtemp,f         ;
        swapf   wtemp,w         ;
        retfie                  ;
```

```
;------------------------------------------------- Subrutina check ------------------------------------------------

check   movlw   01h             ;Segmento
        subwf   reg0,w          ;
        bcf     seg,0           ;Se asume seg = 0
        btfss   status,2        ;Si reg0 = 1 -> seg = 0
        bsf     seg,0           ;

great8  movlw   08h             ;
        subwf   reg1,w          ;Reg1 > 8?
        btfss   status,0        ;Si reg1 <= 8 -> Rotación de store1
        goto    rotar1          ;
        movlw   08h             ;
        subwf   reg1,w          ;
        btfss   status,2        ;
        goto    great16         ;

rotar1  bcf     status,0        ;
        btfsc   seg,0           ;Carry = seg
        bsf     status,0        ;
        rrf     store1,f        ;Rotación de store1 a la der. con carry
        goto    salir           ;

great16 movlw   10h             ;
        subwf   reg1,w          ;Reg1 > 16?
        btfss   status,0        ;Si reg1 <= 16 -> Rotación de store2
        goto    rotar2          ;
        movlw   10h             ;
        subwf   reg1,w          ;
        btfss   status,2        ;
        goto    equal24         ;

rotar2  bcf     status,0        ;
        btfsc   seg,0           ;Carry = seg
        bsf     status,0        ;
        rrf     store2,f        ;Rotación de store2 a la der. con carry
        goto    salir           ;

equal24 movlw   18h             ;
        subwf   reg1,w          ;Reg1 = 24?
        btfsc   status,2        ;Si reg1 = 24 -> Ultima rotación de store3
        goto    rotfin          ;

rotar3  bcf     status,0        ;
        btfsc   seg,0           ;Carry = seg
        bsf     status,0        ;
        rrf     store3,f        ;Rotación de store3 a la der. con carry
        goto    salir           ;

salir   clrf    reg0            ;Inicio del próximo segmento
        return                  ;Retorno

rotfin  bcf     status,0        ;
        btfsc   seg,0           ;Carry = seg
        bsf     status,0        ;
        rrf     store3,f        ;Ultima rotación de store3 a la der.

cero    movlw   1eh             ;Si en el registro store1 tiene una secuencia
        andwf   store1,w        ;igual a 1eh, si store2 tiene la secuencia e0h
        movwf   temp            ;y store3 tiene la secuencia 01h el bit leído
```

```
        movlw   1eh             ;tiene una frecuencia de 2,4 Khz.
        subwf   temp,w          ;
        btfss   status,2        ;
        goto    uno             ;

        movlw   0e0h            ;
        andwf   store2,w        ;
        movwf   temp            ;
        movlw   0e0h            ;
        subwf   temp,w          ;
        btfss   status,2        ;
        goto    fail            ;

        movlw   01h             ;
        andwf   store3,w        ;
        movwf   temp            ;
        movlw   01h             ;
        subwf   temp,w          ;
        btfss   status,2        ;
        goto    fail            ;
        bcf     buffer,tx       ;Conversión a RS-232 para un cero
        goto    next            ;

uno     movlw   06h             ;Si los registros store1, store2 y store3
        andwf   store1,w        ;tienen secuencias iguales a 06h el bit
        movwf   temp            ;leido tiene una frecuencia de 3,6 Khz.
        movlw   06h             ;
        subwf   temp,w          ;
        btfss   status,2        ;
        goto    fail            ;

        movlw   06h             ;
        andwf   store2,w        ;
        movwf   temp            ;
        movlw   06h             ;
        subwf   temp,w          ;
        btfss   status,2        ;
        goto    fail            ;

        movlw   06h             ;
        andwf   store3,w        ;
        movwf   temp            ;
        movlw   06h             ;
        subwf   temp,w          ;
        btfss   status,2        ;
        goto    fail            ;
        bsf     buffer,tx       ;Conversión a RS-232 para un uno

next    incf    reg2,f          ;
        clrf    reg0            ;Iniciación de bit
        clrf    reg1            ;
        clrf    store1          ;
        clrf    store2          ;
        clrf    store3          ;
        return                  ;Retorno
fail    bsf     buffer,err      ;Aviso de error si el bit leído no corresponde
        clrf    tiempo          ;a un uno o un cero en FSK
        clrf    regtemp         ;
        clrf    regcon0         ;
        return                  ;Retorno
```

```
;---------------------------------------------------- Subrutina stop ----------------------------------------------------

stop    bcf     buffer,err      ;Se asume error =  0
        btfsc   buffer,tx       ;Chequeo del bit 10 para definir si hay error
        goto    valido          ;tx = 1 para el bit 10, stop valido
        bsf     buffer,err      ;tx = 0 para el bit 10, stop no valido
        clrf    tiempo          ;Se inicia de nuevo el tiempo de espera
        clrf    regtemp         ;
valido  clrf    regcon0         ;
        clrf    reg0            ;
        clrf    reg1            ;
        clrf    reg2            ;
        movlw   04h             ;Final de la trama, se permite que se vuelva a
        xorwf   buffer,f        ;sincronizar con el osciloscopio (bit 2 del puertob)
        return                  ;

;---------------------------------------------------- Subrutina control ----------------------------------------------------

control incf    reg0,f          ;
        movf    regcon0,w       ;Chequea si regcon0 = 0
        btfsc   status,2        ;
        call    sinc            ;Regcon0 = 0 -> Sincronización con Start bit
        return                  ;

;---------------------------------------------------- Sincronización ----------------------------------------------------

sinc    movlw   04h             ;Señal de sincronización del osciloscopio
        xorwf   buffer,f        ;con el inicio de la trama (bit 2 del puertob)
        clrf    reg0            ;Sincronización con el start bit
        clrf    reg2            ;
        movlw   06h             ;
        movwf   reg1            ;
        movlw   0fch            ;
        movwf   store1          ;
        incf    reg0,f          ;
        bsf     regcon0,0       ;Se deshabilitan futuras sincronizaciones
        return                  ;por un 0. Se restablece con el Stop bit

;-------------------------------------------------------------------------------------------------------------------------
        end
```

8.9 MODULO PSK

El objetivo de este módulo es transmitir una señal digital utilizando modulación PSK, este tipo de modulación consiste en desplazar la fase de una señal portadora de 0º a 180º de acuerdo a la entrada digital de datos.

El primer paso en la conversión a PSK es realizar un muestreo de la línea de entrada, esta operación es llevada a cabo con la ayuda del Timer 0 que se encarga de generar una interrupción a intervalos de tiempo constantes (41,6μs) y durante la cual se toma una muestra del estado de la línea, que es la que contiene la información digital en forma serial.

Al acumularse un total de veinte (20) muestras, es completado un periodo de bit (832μs) y se inicia una operación de procesamiento de las mismas, es decir, se define el estado lógico que tuvo la línea, se realiza la modulación correspondiente al bit que haya sido leído y se identifica si se presentó el inicio o el final de una trama (*Start bit* o *Stop bit*), es decir la posición del bit leído dentro de la trama.

La señal de salida correspondiente al bit modulado hace también uso de la interrupción del Timer 0, ya que este también tiene la función de actualizar el estado de la línea de transmisión entre los módulos, esta actualización se hace en base al contenido de un BUFFER, el cual es un registro donde se almacenan los cambios de estado que se deben hacer a la línea entre los módulos.

Para generar la señal de salida del transmisor, se desplaza la fase de una onda cuadrada (señal portadora) con una frecuencia de 4,8Khz, para formar la onda cuadrada se cambia de estado lógico la señal de salida cada cinco interrupciones, lo anterior se puede apreciar en la Figura 28.

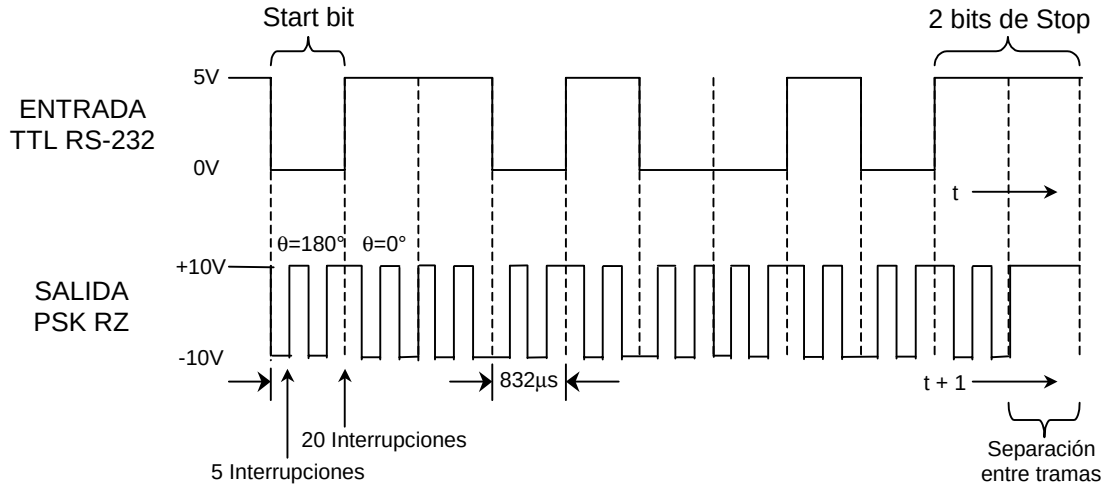


Figura 28 *Formas de onda de entrada RS-232 y salida PSK*

Se observa en la gráfica que un cero lógico es convertido a una onda cuadrada con una fase de 180° y un uno lógico es convertido a una onda cuadrada con una fase de 0°. La salida PSK del módulo esta atrasada un periodo de bit (832µs), con respecto a la señal de entrada, esto se debe al tiempo de procesamiento de las veinte muestras.

La trama de datos utiliza dos bits de parada (*Stop bit*), debido a que se decidió dejar una separación entre tramas de mínimo un periodo de bit para lo cual se usa el segundo bit de parada, esto le permite al módulo detectar el bit de inicio de una próxima trama. De los dos bits sólo el primero se convierte a PSK, el segundo tendrá un estado lógico uno.

En el módulo de recepción se decodifica la señal PSK y se recupera la trama de datos.

Para realizar la demodulación se aprovecha que existen dos señales de diferente fase que representan los estados lógicos cero y uno, por lo que el receptor cumple la función de detector de fase para determinar el bit que se debe recuperar a la salida.

En resumen el proceso usado para la transmisión y recepción del módulo PSK se puede describir como se aprecia en la Figura 29:

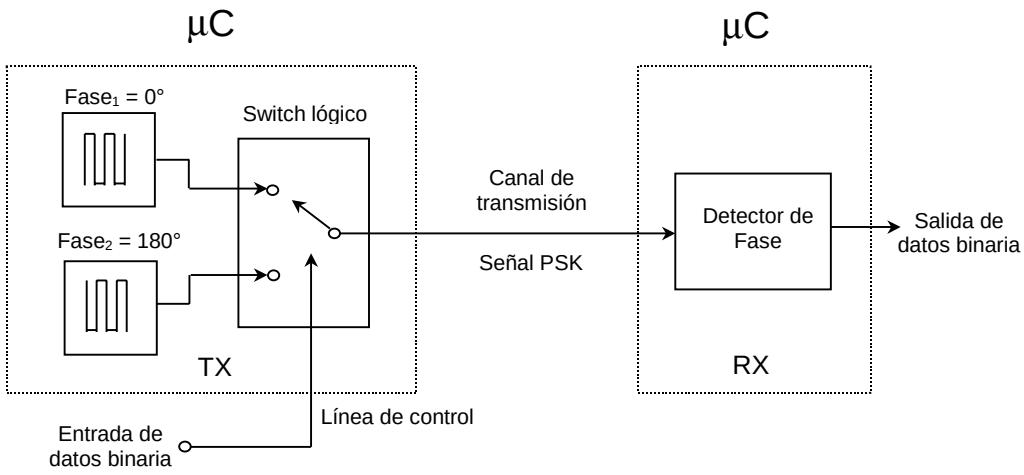


Figura 29 *Diagrama de bloques de los módulos de transmisión y recepción PSK*

donde el transmisor realiza un proceso que equivale a conmutar la línea de salida entre dos osciladores, uno en desfase de 180° con respecto al otro, la conmutación se hace de acuerdo a la entrada de datos binaria. En la etapa de recepción el detector de fase se encarga de recuperar los datos binarios y enviarlos a la salida.

8.9.1 Software del transmisor. El software del transmisor es quien realiza la modulación PSK de la entrada digital. La información serial de la entrada digital, la cual es entregada por un Computador, es llevada al microcontrolador por medio de un driver de línea (MAX232), este se encarga de llevar la señal a niveles de voltaje TTL, correspondientes a los niveles de voltaje que maneja el microcontrolador.

Al inicio del programa principal, se setea el bit seis del puerto B (RB6), el cual entrega la señal de salida modulada, esto se hace para indicar que la salida se encuentra en estado de reposo. Luego el programa ingresa a un loop infinito, el cual solo es interrumpido por el desbordamiento del Timer 0 que genera una interrupción. Esta interrupción dirige el programa a la dirección de memoria 0004H donde se encuentra el vector de interrupciones. En este sitio se ubica un salto que apunta a la dirección donde se encuentra la rutina de servicio a la interrupción. Esta rutina es la que finalmente se encarga del procesamiento de la entrada digital (bit siete del puerto B). La rutina de servicio a la interrupción es la encargada de realizar el muestreo de la entrada digital y de generar la señal de salida. Al iniciarse esta rutina se entra a la subrutina Tiempo de espera, la cual realiza un retardo de aproximadamente 150ms, la línea de entrada debe de estar en estado de reposo (uno lógico) durante el tiempo de espera para continuar con el programa, el caso contrario indicara que al momento de iniciarse el programa el Computador estaba realizando una transmisión y esta será descartada. Luego del tiempo de espera el programa queda listo para recibir el *Start bit* de una trama.

Al instante en que se detecta un *Start bit* se empieza a contar veinte interrupciones (duración de un bit), durante las cuales se lee el estado lógico de la entrada, al completarse estas veinte interrupciones se entra a la subrutina Txout donde se verifica el estado lógico del bit leído y se decide la fase que deberá el bit modulado. Este proceso de conteo de veinte interrupciones se repite con los ocho bits de datos y el *Stop bit*. Posterior a la verificación hecha por la subrutina Txout se realiza un conteo de interrupciones durante los cuales se harán cambios de estado para formar la señal de onda cuadrada con la fase indicada.

Al terminar el muestreo y la modulación de la trama completa se verifica que el ultimo bit leído sea un *Stop bit* (uno lógico), si es así se realiza un reinicio del programa, es decir, se vuelve a las condiciones de espera de un próximo *Start bit*, pero en el caso en que el ultimo bit leído sea un cero lógico, significara que hay un error de transmisión, por lo cual se dará un aviso de error y se entrara de nuevo a la subrutina Tiempo de espera.

A continuación se hace una descripción detallada de las rutinas implementadas para el desarrollo del modulador PSK.

8.9.1.1 Programa principal. En esta parte del programa se configura el Timer 0, el cual utiliza la fuente de reloj interna del microcontrolador, se programan el bit siete del puerto B como entrada y los bits cero a seis del puerto B como salida, se inicializan todos los registros que se utilizan en el programa, se

habilita la interrupción del Timer 0 y se setea el bit seis del puerto B para indicar que la salida se encuentra en estado de reposo. Esto es realizado únicamente al inicio del programa y no se vuelve a entrar en el mientras el programa este corriendo. Luego el programa ingresa a un loop infinito, es decir, se ejecuta un salto que apunta a la dirección de memoria donde se encuentra el mismo salto, este lazo cerrado que se produce solo se interrumpe por la acción de desbordamiento del Timer 0 cada 41,6μs para procesar la entrada digital.

8.9.1.2 Rutina de servicio a la interrupción del Timer 0. Esta rutina inicialmente ejecuta un PUSH, el cual se encarga de almacenar el registro de trabajo W y las banderas de estado de la ALU como Carry, Digit Carry y Cero que se encuentran en el registro STATUS. Luego se realiza el ajuste de tiempo del Timer 0, este ajuste se realiza para cargar el Timer 0 correctamente y de esta manera calibrar el tiempo en que se presentara la próxima interrupción.

A continuación se realiza una actualización del puerto B que consiste en mandar a las salidas del puerto los cambios que se hayan hecho al registro BUFFER. Después se verifica que la bandera TIEMPO sea uno, esto indica que el tiempo de espera ya ha sido ejecutado, en el caso en que sea cero se salta a la subrutina Tiempo de espera. El registro REGCON0 es el registro encargado de controlar la sincronización del programa con el *Start bit* de una trama, si este registro es uno indicara que ya se presento un *Start bit* por lo

tanto se debe empezar el conteo de las veinte interrupciones, este conteo lo realiza el registro REG1.

Al completarse veinte interrupciones se llama a la subrutina Txout, esta se encarga de verificar el estado lógico del bit leído y decidir la fase que deberá tener la señal modulada de salida. El registro REG2 es quien lleva el conteo del numero de bits leídos, si este es diferente de cero indicara que ya se ha leído el primer bit, por lo tanto se debe empezar la modulación, para realizarla se efectúan cinco cambios de estado durante veinte interrupciones, de esta manera se forma una onda cuadrada de 4,8KHz, la fase de esta onda se ha determinado con anterioridad en la subrutina Txout. Cada cambio de estado se envía al bit seis del registro BUFFER.

Al realizar la lectura del bit siete (RD) del puerto B, se verifica el estado lógico de la entrada digital, si es un cero lógico se llama a la subrutina Control, la cual se encarga de llevar el conteo del numero de ceros lógicos que se han leído y de controlar la sincronización del programa con el *Start bit*.

Si el registro REG2 es igual a diez indicara que ya se ha leído la trama completa, por lo que se llama a la subrutina Stop, que se encarga de verificar que el ultimo bit leído sea un *Stop bit* valido (uno lógico). Al haberse completado la lectura de la trama completa se produce un reinicio del programa, es decir, se prepara al programa para detectar un próximo *Start bit*, pero este reinicio se ejecuta quince interrupciones después de que se ha leído

el *Stop bit*, debido a que se debe permitir que este ultimo termine de ser modulado, ya que la modulación se hace después de realizar la lectura.

Por ultimo se ejecuta un POP para restaurar el registro W y las banderas de Carry, Digit Carry y Cero que se habían almacenado a su registro original (STATUS).

8.9.1.3 Subrutina Tiempo de espera. La subrutina de tiempo de espera consiste en un retardo de 150ms, esto se logra contando las interrupciones del Timer 0 que se producen cada 41,6μs, de esta manera se realiza el conteo de 256 interrupciones lo que proporciona un tiempo de 10,7ms, para lograr un tiempo de 150ms se cuentan 17 ciclos de 10,7ms.

Mientras se ejecuta el tiempo de espera se realiza una lectura de la entrada digital, si el resultado de la lectura es un uno lógico (estado de reposo) se sigue con el conteo hasta completar el tiempo de espera requerido, después de cumplido esto el registro TIEMPO, el cual actúa como bandera para informar que el tiempo de espera ha sido ejecutado, se carga con un uno y la bandera de ERROR (bit cinco del puerto B) se clarea, pero si la lectura da como resultado un cero lógico indicara que una transmisión ha comenzado antes de dar inicio al programa y esto provocara un error de transmisión, lo cual hace que se reinicie de nuevo el tiempo de espera y se setea la bandera de ERROR.

8.9.1.4 Subrutina Txout. Esta subrutina revisa el registro REG0, el cual contiene el numero de lecturas que correspondieron a un cero lógico durante las veinte interrupciones, si de las veinte lecturas realizadas once o más fueron un cero lógico indicara que el bit leído es un cero, pero si de las veinte lecturas tomadas diez o menos fueron un cero lógico quiere decir que el bit leído es un uno. De esta manera se realiza un promedio de las lecturas para descartar los posibles cambios en la señal debidos al ruido que se pueda presentar.

Si el bit leído es un cero, la señal de salida deberá tener una fase de 0 grados, de esta manera la señal modulada tendrá un estado inicial alto (uno lógico), pero si el bit leído es un uno, la señal de salida deberá tener una fase de 180 grados, para lo cual la señal modulada tendrá un estado inicial bajo (cero lógico). El estado inicial de la onda cuadrada es puesto en el bit seis (TX) del registro BUFFER. El resultado de las veinte lecturas, es decir, uno o cero lógico, es almacenado en el registro REGAUX. Luego se prepara al programa para leer el próximo bit.

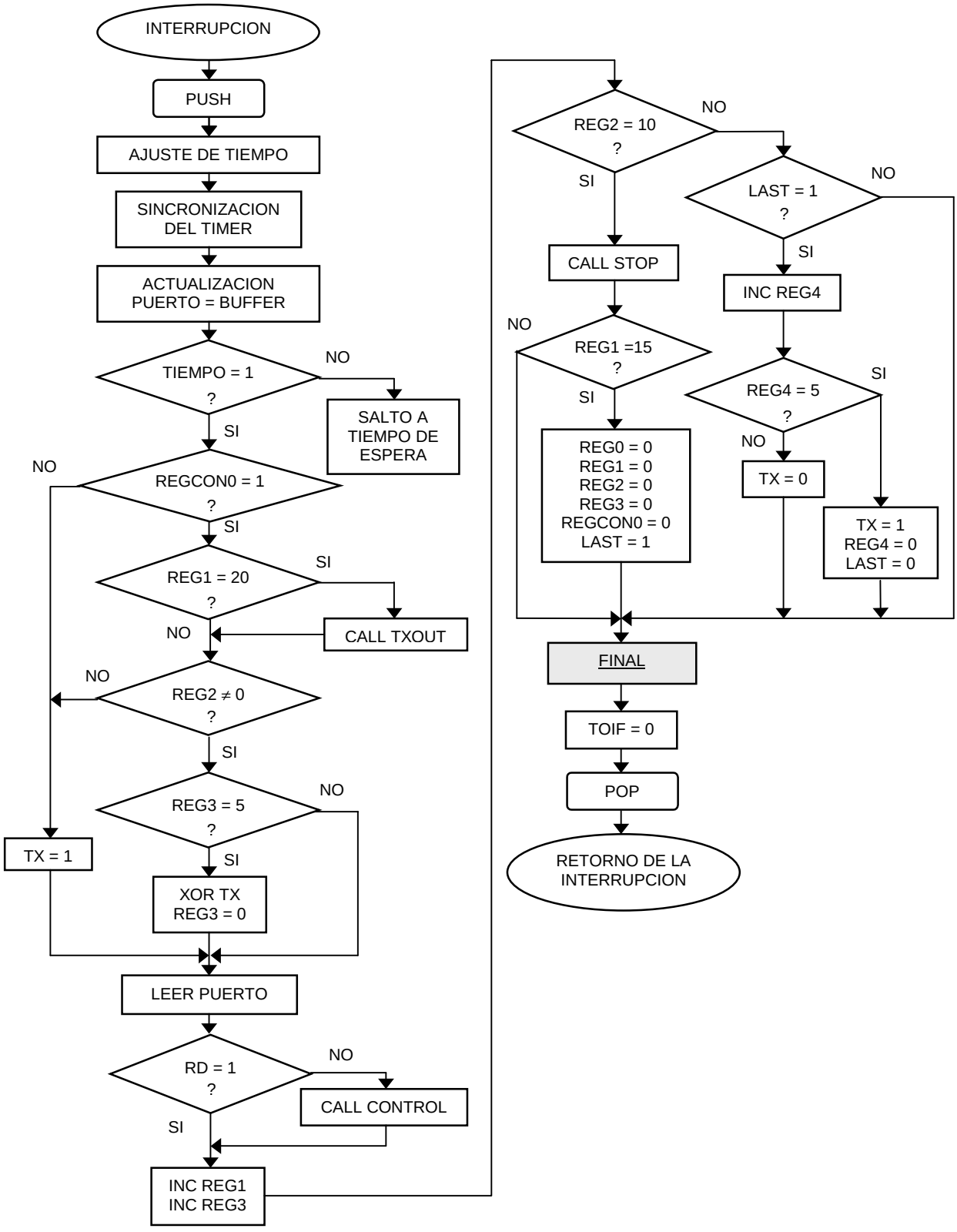
8.9.1.5 Subrutina Control. Esta subrutina es quien realiza el conteo de las lecturas que corresponden a un cero lógico, incrementando el registro REG0. También realiza la sincronización del programa con el *Start bit* de una trama, esta sincronización consiste en preparar al programa para empezar la lectura de todos los bits de la trama, esto se logra inicializando los registros encargados de este proceso. La subrutina Control utiliza el registro REGCON0 como bandera de control para indicar que la sincronización ya se efectúo, la

sincronización solo se hace cuando se detecta el *Start bit* y no se realiza mientras se lee la trama completa, después de que esta ha sido leída y modulada el programa queda a la espera de un nuevo *Start bit* y de una nueva sincronización.

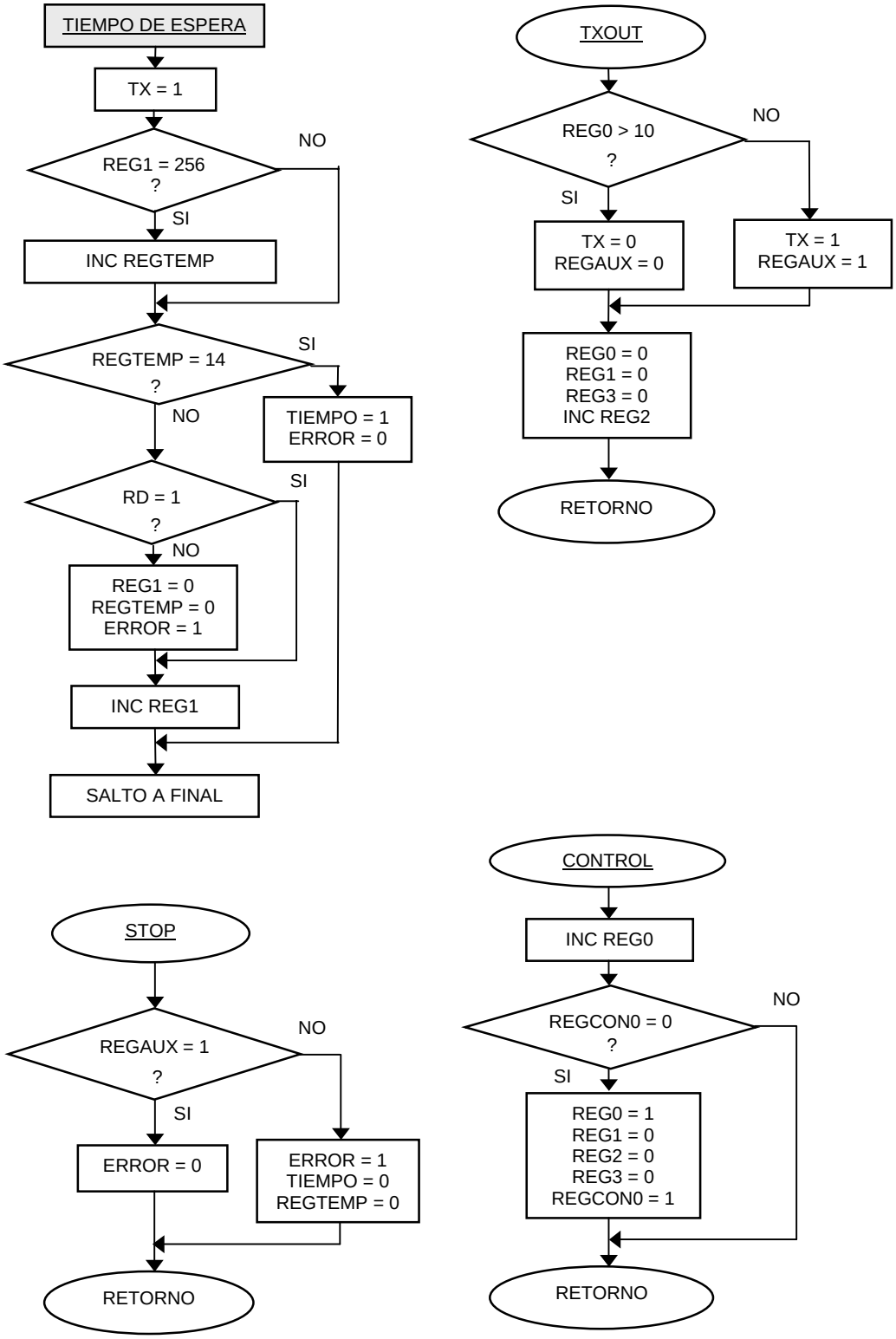
8.9.1.6 Subrutina Stop. Esta subrutina consiste en verificar si el ultimo bit leído (décimo bit) corresponde a un *Stop bit* valido, es decir, un uno lógico. La verificación se realiza revisando el contenido del registro REGAUX, el cual almacena el ultimo bit leído. Si el registro REGAUX es un uno, indicara que se leyó un *Stop bit* que es valido, de esta manera se clarea la bandera de ERROR y se retorna, pero en el caso contrario, es decir, si el registro REGAUX es cero indica que el *Stop bit* leído no es valido causando un error de comunicación, para lo cual se setea la bandera de ERROR y se clarea la bandera de TIEMPO obligando al programa a entrar de nuevo a la subrutina Tiempo de espera.

8.9.2 Diagrama de flujo del transmisor. A continuación se presenta el diagrama de flujo de la rutina de servicio a la interrupción y de las subrutinas que fueron necesarias implementar en el desarrollo de la etapa de transmisión para el módulo PSK.

RUTINA DE SERVICIO A LA INTERRUPCION DEL TRANSMISOR PSK



SUBRUTINAS


TIEMPO DE ESPERA
TX = 1
REG1 = 256 ?
NO
SI
INC REGTEMP
REGTEMP = 14 ?
SI
NO
TIEMPO = 1
ERROR = 0
RD = 1 ?
SI
NO
REG1 = 0
REGTEMP = 0
ERROR = 1
INC REG1
SALTO A FINAL
TXOUT
REG0 > 10 ?
NO
SI
TX = 0
REGAUX = 0
TX = 1
REGAUX = 1
REG0 = 0
REG1 = 0
REG3 = 0
INC REG2
RETORNO
STOP
REGAUX = 1 ?
NO
SI
ERROR = 0
ERROR = 1
TIEMPO = 0
REGTEMP = 0
RETORNO
CONTROL
INC REG0
REGCON0 = 0 ?
NO
SI
REG0 = 1
REG1 = 0
REG2 = 0
REG3 = 0
REGCON0 = 1
RETORNO

8.9.3 Programa en código fuente del transmisor. A continuación se da el programa detallado, que se utilizo en el microcontrolador, con sus respectivos comentarios.

```
;-------------------------------------------------------------------------------------------------------------------
;                                        Modulador PSK (Transmisor)
;-------------------------------------------------------------------------------------------------------------------
;       PSKTX.ASM

        list p=16c622           ;Procesador
        include <p16c622.inc>

;       Al programar el micro tener en cuenta los fusibles de configuración
;       OSC = HS, WDT = OFF, PWRTE = OFF, CP = OFF, BODEN = OFF

;------------------------------------------------- Zona de Registros ---------------------------------------------

tmr_opt     equ     01h     ;Timer0 / option
status      equ     03h     ;
intcon      equ     0bh     ;
puertob     equ     06h     ;
buffer      equ     2ah     ;Registro que almacena cambios del puertob
reg0        equ     2bh     ;Contador de ceros
reg1        equ     2ch     ;Contador de interrupciones
reg2        equ     2dh     ;Contador de bits
reg3        equ     2eh     ;Contador de paquetes de 5 interrupciones
reg4        equ     2fh     ;Contador de las ultimas 5 interrupciones de una trama
reg5        equ     30h     ;Contador de interrupciones para los osciladores
regaux      equ     31h     ;Registro auxiliar que almacena el contenido de tx
regcon0     equ     32h     ;Bandera de control de start bit
tempwr      equ     33h     ;Temporal del reloj
tiempo      equ     34h     ;Bandera de control del tiempo de espera
regtemp     equ     35h     ;Contador para determinar el tiempo de espera
last        equ     36h     ;Bandera para habilitar tx en las 5 int. finales
wtemp       equ     70h     ;Temporal de w
stemp       equ     71h     ;Temporal de status

;------------------------------------------------- Asignaciones de bit -------------------------------------------

rd          equ     7       ;Entrada serial (RS-232)
tx          equ     6       ;Salida serial (PSK)
err         equ     5       ;Aviso de error
gie         equ     7       ;Habilita int. general
toie        equ     5       ;Habilita int. por tmr0
toif        equ     2       ;Flag de int. por tmr0
rp0         equ     5       ;Página de memoria
osc1        equ     4       ;Portadora
osc2        equ     3       ;Portadora desplazada
sincro      equ     2       ;Sincronización del osciloscopio
```

```
;------------------------------------------------ Configuración del PIC ------------------------------------------------

        org     0
        goto    inicio          ;Inicio del programa principal

        org     4
        goto    inter           ;Vector de interrupciones

        org     5               ;Inicio programa "inicio"

inicio  bsf     status,rp0      ;Va al banco1
        movlw   08h             ;Option, clock interno
        movwf   tmr_opt         ;
        movlw   80h             ;Configuración de puertos
        movwf   puertob         ;Puertob 7:entrada, 6-0:salida
        bcf     status,rp0      ;Retorno al banco0

        clrf    tmr_opt         ;Inicialización de registros
        clrf    puertob         ;
        clrf    reg0            ;
        clrf    reg1            ;
        clrf    reg2            ;
        clrf    reg3            ;
        clrf    reg4            ;
        clrf    reg5            ;
        clrf    regaux          ;
        clrf    regcon0         ;
        clrf    buffer          ;
        clrf    tempwr          ;
        clrf    tiempo          ;
        clrf    regtemp         ;
        clrf    last            ;

        bsf     intcon,gie      ;Habilitación general de interrupciones
        bsf     intcon,toie     ;Habilita interrupción de timer0
        bsf     buffer,tx       ;Inicialización de salida serial (tx)
        bsf     buffer,osc1     ;Inicialización de señales generadas
        bcf     buffer,osc2     ;
        bcf     buffer,sincro   ;

;------------------------------------------------------------ Loop ------------------------------------------------------------

loop    goto    loop            ;Programa principal, loop infinito


;------------------------ RUTINA DE SERVICIO A LA INTERRUPCION DEL TMRO ----------------------


inter   movwf   wtemp           ;Push
        swapf   status,w        ;
        movwf   stemp           ;

        movf    tmr_opt,w       ;Ajuste de tiempo del timer0
        movwf   tempwr          ;
        btfss   tempwr,0        ;
        goto    time3e          ;

time3e  movlw   3eh             ;3eh -> 41,6 useg (aproximadamente)
        movwf   tmr_opt         ;Carga del timer0
```

```
princip  movf    buffer,w        ;
         movwf   puertob         ;Actualización del puerto b

         btfss   tiempo,0        ;Se verifica si ya se ejecuto el tiempo de espera
         goto    time            ;

         btfss   regcon0,0       ;Mientras no se de un start bit, tx es 1 (reposo)
         goto    jump            ;

;---------------------------------------------------------- Conversión ----------------------------------------------------------


         movlw   14h             ;
         subwf   reg1,w          ;Reg1 = 20?
         btfsc   status,2        ;Si reg1 = 20 -> call txout
         call    txout           ;Actualización de tx

         movf    reg2,w          ;Si reg2 es diferente de cero se permite
         btfsc   status,2        ;la conversión del primer bit leído
         goto    jump            ;

         movlw   05h             ;Se cuentan 5 interrupciones
         subwf   reg3,w          ;
         btfss   status,2        ;
         goto    leer            ;

         movlw   40h             ;XOR de tx cada 5 interrupciones
         xorwf   buffer,f        ;
         clrf    reg3            ;
         goto    leer            ;

jump     bsf     buffer,tx       ;Estado de reposo de la línea.


;---------------------------------------------------------- Muestreo ----------------------------------------------------------


leer     btfss   puertob,rd      ;Lectura de la entrada serial
         call    control         ;
         incf    reg1,f          ;Incrementa contador de interrupciones
         incf    reg3,f          ;Incrementa contador de paquetes de 5 interrupciones
         incf    reg5,f          ;Incrementa contador de osciladores


;---------------------------------------------------------- Stop ----------------------------------------------------------


stopbit  movlw   0ah             ;Verificación del stop (bit 10)
         subwf   reg2,w          ;
         btfss   status,2        ;reg2 = 10?
         goto    ultimo          ;
         bcf     buffer,err      ;Se asume error = 0
         btfsc   regaux,0        ;Chequeo del bit 10 para definir si hay error
         goto    valido          ;Tx = 1 para el bit 10, stop valido
         bsf     buffer,err      ;Tx = 0 para el bit 10, stop no valido
         clrf    tiempo          ;Se inicia de nuevo el tiempo de espera
         clrf    regtemp         ;
         goto    final           ;
```

```
;------------------------------------------------------- Reset ---------------------------------------------------

valido  movlw   0fh             ;El Reset se da en la interrupción 15 después
        subwf   reg1,w          ;de que se ha leído el stop bit
        btfss   status,2        ;
        goto    final           ;
        clrf    regcon0         ;Reset
        clrf    reg0            ;
        clrf    reg1            ;
        clrf    reg2            ;
        clrf    reg3            ;
        bsf     last,0          ;Habilita la salida de tx en las 5 int. finales
        goto    final           ;


;------------------------------------------------------- Last ----------------------------------------------------

ultimo  btfss   last,0          ;
        goto    final           ;

        incf    reg4,f          ;Conteo de las 5 interrupciones faltantes
        movlw   05h             ;
        subwf   reg4,w          ;
        btfss   status,2        ;
        goto    recover         ;

        clrf    last            ;Finalización de la recuperación del 4 segmento
        clrf    reg4            ;
        movlw   04h             ;Final de la trama, se permite que se vuelva a
        xorwf   buffer,f        ;sincronizar con el osciloscopio (bit 2 del puertob)
        bsf     buffer,tx       ;
        goto    final           ;

recover bcf     buffer,tx       ;Recuperación del cuarto segmento
        goto    final           ;


;------------------------------------------- Subrutina tiempo de espera ----------------------------------------


time    bsf     buffer,tx       ;Mientras se ejecuta el tiempo de espera, tx es 1 (reposo)
        movlw   0ffh            ;Tiempo de espera de aproximadamente 150mseg
        subwf   reg1,w          ;41.6useg * 256 = 10.6mseg
        btfsc   status,2        ;
        incf    regtemp,f       ;Se cuentan ciclos de 10.6mseg
        movlw   0eh             ;
        subwf   regtemp,w       ;10.6mseg * 14 = 149.1mseg
        btfss   status,2        Regtemp = 14 -> tiempo = 1 y error = 0
        goto    again           ;
        bsf     tiempo,0        ;Cumplidos los 150mseg se setea la bandera de tiempo
        bcf     buffer,err      ;Se clarea la bandera de error
        goto    final           ;

again   btfsc   puertob,rd      ;Lectura de la entrada serial
        goto    uno             ;
        clrf    reg1            ;Si rd es cero durante el tiempo de espera
        clrf    regtemp         ;se inicia de nuevo el conteo
        bsf     buffer,err      ;Aviso de error por transmisión en proceso
uno     incf    reg1,f          ;Incremento del contador de interrupciones
```

```
;---------------------------------------------------------- Final ----------------------------------------------------------

final   movlw   05h             ;Generación de la portadora y de su
        subwf   reg5,w          ;desplazamiento en fase (bits 3 y 4 del puertob)
        btfss   status,2        ;
        goto    salir           ;
        movlw   18h             ;
        xorwf   buffer,f        ;
        clrf    reg5            ;

salir   bcf     intcon,toif     ;Se clarea la bandera del tmr0
        swapf   stemp,w         ;Pop
        movwf   status          ;
        swapf   wtemp,f         ;
        swapf   wtemp,w         ;
        retfie                  ;

;------------------------------------------------- Subrutina txout ----------------------------------------------------

txout   movf    reg0,w          ;
        sublw   0ah             ;(10 - reg0)
        btfss   status,0        ;Si reg0 > 10 -> tx = 0
        goto    tx0             ;
tx1     bsf     buffer,tx       ;tx = 1
        bsf     regaux,0        ;Se almacena bit leído en regaux
        goto    done            ;

tx0     bcf     buffer,tx       ;tx = 0
        bcf     regaux,0        ;Se almacena bit leído en regaux

done    clrf    reg0            ;Iniciación de bit
        clrf    reg1            ;
        incf    reg2,f          ;
        clrf    reg3            ;
        return                  ;

;------------------------------------------------- Subrutina control --------------------------------------------------

control incf    reg0,f          ;
        movf    regcon0,w       ;Chequea si regcon0 = 0
        btfsc   status,2;
        call    sinc            ;Regcon0 = 0 -> Sincronización con Start bit
        return                  ;

;------------------------------------------------- Sincronización -----------------------------------------------------

sinc    movlw   04h             ;Señal de sincronización del osciloscopio
        xorwf   buffer,f        ;con el inicio de la trama (bit 2 del puertob)
        clrf    reg0            ;Sincronización con el start bit
        clrf    reg1            ;
        clrf    reg2            ;
        clrf    reg3            ;
        incf    reg0,f          ;
        bsf     regcon0,0       ;Se deshabilitan futuras sincronizaciones
        return                  ;por un bit 0. Se restablece con el Stop bit

;------------------------------------------------------------------------------------------------------------------------
        end
```

8.9.4 Software del receptor. El software del receptor es el encargado de demodular la señal PSK y recuperar la información digital. Este software divide el bit modulado en cuatro segmentos, cada segmento esta compuesto por cinco interrupciones, durante las cuales se realiza la lectura de la entrada modulada, al final de estas cinco interrupciones se define el estado lógico del segmento leído y se almacena en el registro STORE. Al terminar de leer el bit modulado, en el registro STORE quedara guardada la secuencia de segmentos y esta ayudara a determinar si la fase del bit modulado es 0° o 180°.

Al inicio del programa principal se setea el bit seis del puerto B (RB6), el cual entrega la señal de salida demodulada, esto se hace para indicar que la salida se encuentra en estado de reposo. Luego el programa ingresa a un loop infinito, el cual solo es interrumpido por el desbordamiento del Timer 0 que genera una interrupción. Esta interrupción dirige el programa a la dirección de memoria 0004H donde se encuentra el vector de interrupciones. En este sitio se ubica un salto que apunta a la dirección donde se encuentra la rutina de servicio a la interrupción. Es esta rutina la que finalmente se encarga del procesamiento de la entrada modulada (bit siete del puerto B).

La rutina de servicio a la interrupción es la encargada de realizar el muestreo de la entrada modulada y de generar la señal de salida. Al iniciarse esta rutina se entra a la subrutina Tiempo de espera, la cual realiza un retardo de aproximadamente 150ms en el cual se verifica que el Computador no este transmitiendo al momento de iniciar el programa.

Luego de ejecutar el tiempo de espera el programa queda listo para recibir el *Start bit* modulado de una trama. Al instante en que se detecta un *Start bit* se empieza a contar cinco interrupciones, durante las cuales se lee el estado lógico de la entrada, al completarse estas cinco interrupciones (duración de un segmento del bit modulado) se entra a la subrutina Chequeo donde se verifica el estado lógico del segmento leído y se almacena, este proceso es repetido hasta obtener los cuatro segmentos correspondientes a un bit modulado, lo cual tarda veinte interrupciones, al final de las cuales se determina la fase del bit modulado y se define el estado lógico de la salida digital. Los ocho bits de datos y el *Stop bit* son demodulados de la misma forma.

Al terminar el muestreo y la demodulación de la trama completa, se verifica que el ultimo bit leído sea un *Stop bit* valido, si es así se realiza un reinicio del programa, es decir, se vuelve a las condiciones de espera de un próximo *Start bit*, pero en el caso en que el ultimo bit leído sea un cero lógico, significara que hay un error de transmisión, por lo cual se dará un aviso de error y se entrara de nuevo a la subrutina Tiempo de espera.

A continuación se hace una descripción detallada de las rutinas implementadas para el desarrollo del demodulador PSK.

8.9.4.1 Programa principal. En esta parte del programa se configura el Timer 0, el cual utiliza la fuente de reloj interna del microcontrolador, se programan el bit siete del puerto B como entrada y los bits cero a seis del puerto B como

salida, se inicializan todos los registros que se utilizan en el programa, se habilita la interrupción del Timer 0 y se setea el bit seis del puerto B para indicar que la salida se encuentra en estado de reposo. Esto es realizado únicamente al inicio del programa y no se vuelve a entrar en el mientras el programa este corriendo. Luego el programa ingresa a un loop infinito, es decir, se ejecuta un salto que apunta a la dirección de memoria donde se encuentra el mismo salto, este lazo cerrado que se produce solo se interrumpe por la acción de desbordamiento del Timer 0 cada 41,6μs para procesar la entrada modulada.

8.9.4.2 Rutina de servicio a la interrupción del Timer 0. Esta rutina inicialmente ejecuta un PUSH, el cual se encarga de almacenar el registro de trabajo W y las banderas de estado de la ALU. Luego se realiza el ajuste de tiempo del Timer 0, este ajuste se realiza para cargar el Timer 0 correctamente y de esta manera calibrar el tiempo en que se presentara la próxima interrupción.

Posteriormente se realiza una actualización del puerto B que consiste en mandar a las salidas del puerto los cambios que se hayan hecho al registro BUFFER. Después se verifica que la bandera TIEMPO sea uno, esto indica que el tiempo de espera ya ha sido ejecutado, en el caso en que sea cero se salta a la subrutina Tiempo de espera. El registro REGCON0 es el registro encargado de controlar la sincronización del programa con el *Start bit* modulado, si este registro es uno indicara que ya se presento un *Start bit* por lo

tanto se debe empezar el conteo de interrupciones, este conteo lo realiza el registro REG1.

A intervalos de cinco interrupciones se llama a la subrutina Chequeo, esta se encarga de verificar y almacenar el estado lógico del segmento leído, esto se repite hasta completar cuatro segmentos, lo cual toma veinte interrupciones, al completar cuatro segmentos se determina la fase del bit modulado y se define el estado lógico de la salida digital. Este proceso se repite con los ocho bits de datos y el *Stop bit*.

Al realizar la lectura del bit siete (RD) del puerto B, se verifica el estado lógico de la entrada modulada, si es un cero lógico se llama a la subrutina Control, la cual se encarga de llevar el conteo del numero de ceros lógicos que se han leído y de controlar la sincronización del programa con el *Start bit* modulado.

Si el registro REG2 es igual a diez, indicara que se ha leído la trama completa, por lo que se llama a la subrutina Stop, que se encarga de verificar que el ultimo bit leído sea un *Stop bit* valido. Dentro de esta subrutina se produce el reinicio del programa, es decir, se prepara al programa para detectar un próximo *Start bit*.

Por ultimo se ejecuta un POP para restaurar el registro W y las banderas de estado de la ALU a su registro original (STATUS).

8.9.4.3 Subrutina Tiempo de espera. La subrutina de tiempo de espera consiste en un retardo de 150ms, esto se logra contando las interrupciones del Timer 0 que se producen cada 41,6μs, de esta manera se realiza el conteo de 256 interrupciones lo que proporciona un tiempo de 10,7ms, para lograr un tiempo de 150ms se cuentan 14 ciclos de 10,7ms.

Mientras se ejecuta el tiempo de espera se realiza una lectura de la entrada modulada, si el resultado de la lectura es un uno lógico (estado de reposo) se sigue con el conteo hasta completar el tiempo de espera requerido, después de cumplido esto el registro TIEMPO, el cual actúa como bandera para informar que el tiempo de espera ha sido ejecutado, se carga con un uno y la bandera de ERROR (bit cinco del puerto B) se clarea, pero si la lectura da como resultado un cero lógico indicara que una transmisión ha comenzado antes de dar inicio al programa y esto provocara un error de transmisión, lo cual hace que se reinicie de nuevo el tiempo de espera y se setea la bandera de ERROR.

8.9.4.4 Subrutina Chequeo. Esta subrutina revisa el registro REG0 a intervalos de cinco interrupciones, el REG0 contiene el numero de lecturas que correspondieron a un cero lógico durante cinco interrupciones, si de las cinco lecturas realizadas tres o más fueron un cero lógico esto indicara que el segmento leído es un cero, pero si dos o menos de las lecturas tomadas fueron un cero lógico quiere decir que el segmento leído es un uno. De esta manera se realiza un promedio de las lecturas para descartar los posibles cambios en la señal debidos al ruido que se pueda presentar. Se acumulan entonces cuatro

segmentos durante veinte interrupciones, cada uno de los segmentos es almacenado en el registro STORE.

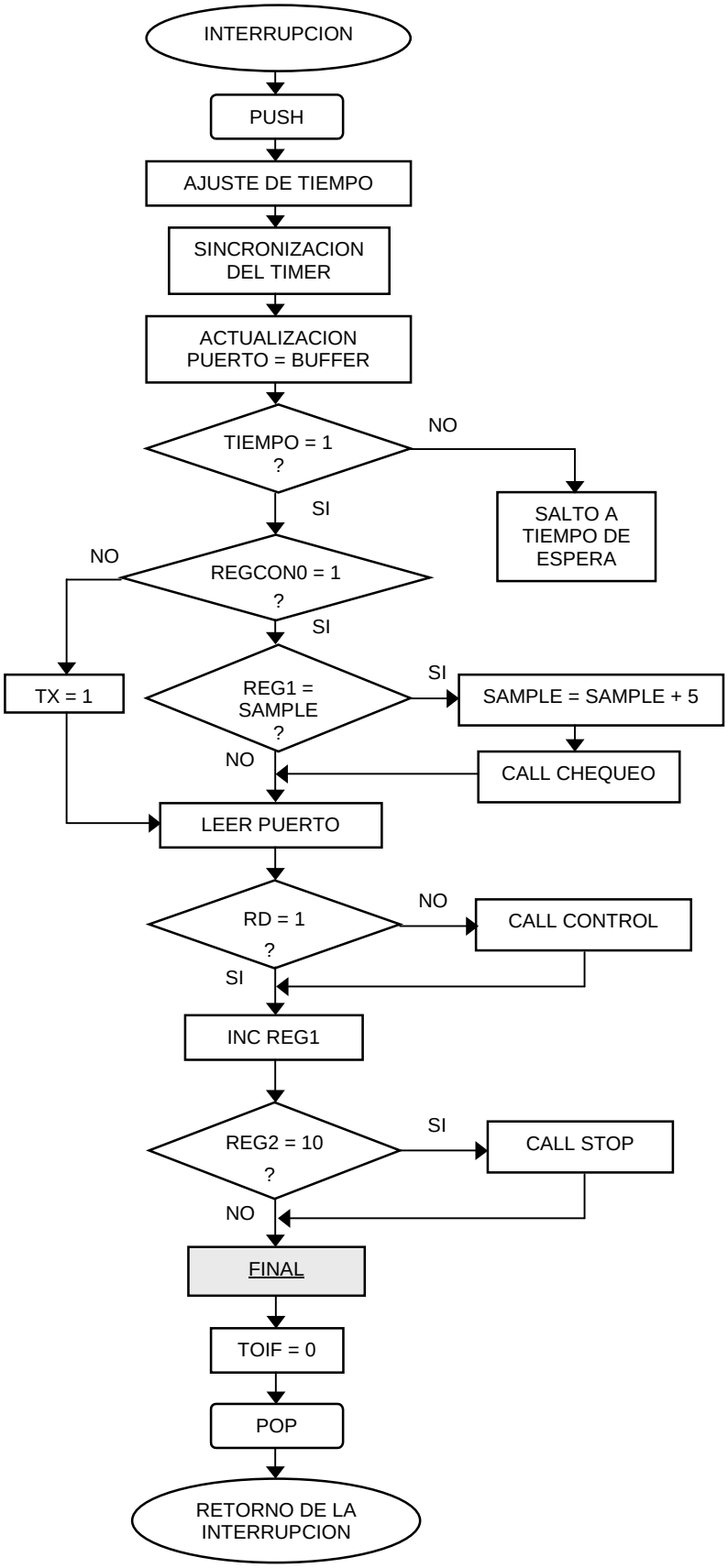
Al completar cuatro segmentos, en el registro STORE queda una secuencia de unos y ceros que ayuda a determinar la fase del bit modulado. Si el registro STORE es igual a 50H indicara que la fase del bit modulado es 0°, por lo tanto la salida digital, por la cual se obtiene el bit demodulado, deberá tener un estado lógico uno, pero si el registro STORE es igual a A0H indicara que la fase del bit modulado es 180° y la salida digital deberá tener un estado lógico cero. Luego se prepara al programa para leer el próximo bit.

8.9.4.5 Subrutina Control. Esta subrutina es quien realiza el conteo de las lecturas que corresponden a un cero lógico, incrementando el registro REG0. También realiza la sincronización del programa con el *Start bit* modulado de una trama, esta sincronización consiste en preparar al programa para empezar la lectura de todos los bits modulados, esto se logra inicializando los registros encargados de este proceso. La subrutina Control utiliza el registro REGCON0 como bandera de control para indicar que la sincronización ya se efectúo, la sincronización solo se hace cuando se detecta el *Start bit* modulado y no se realiza mientras se lee la trama modulada, después de que esta ha sido leída y demodulada el programa queda a la espera de un nuevo *Start bit* y de una nueva sincronización.

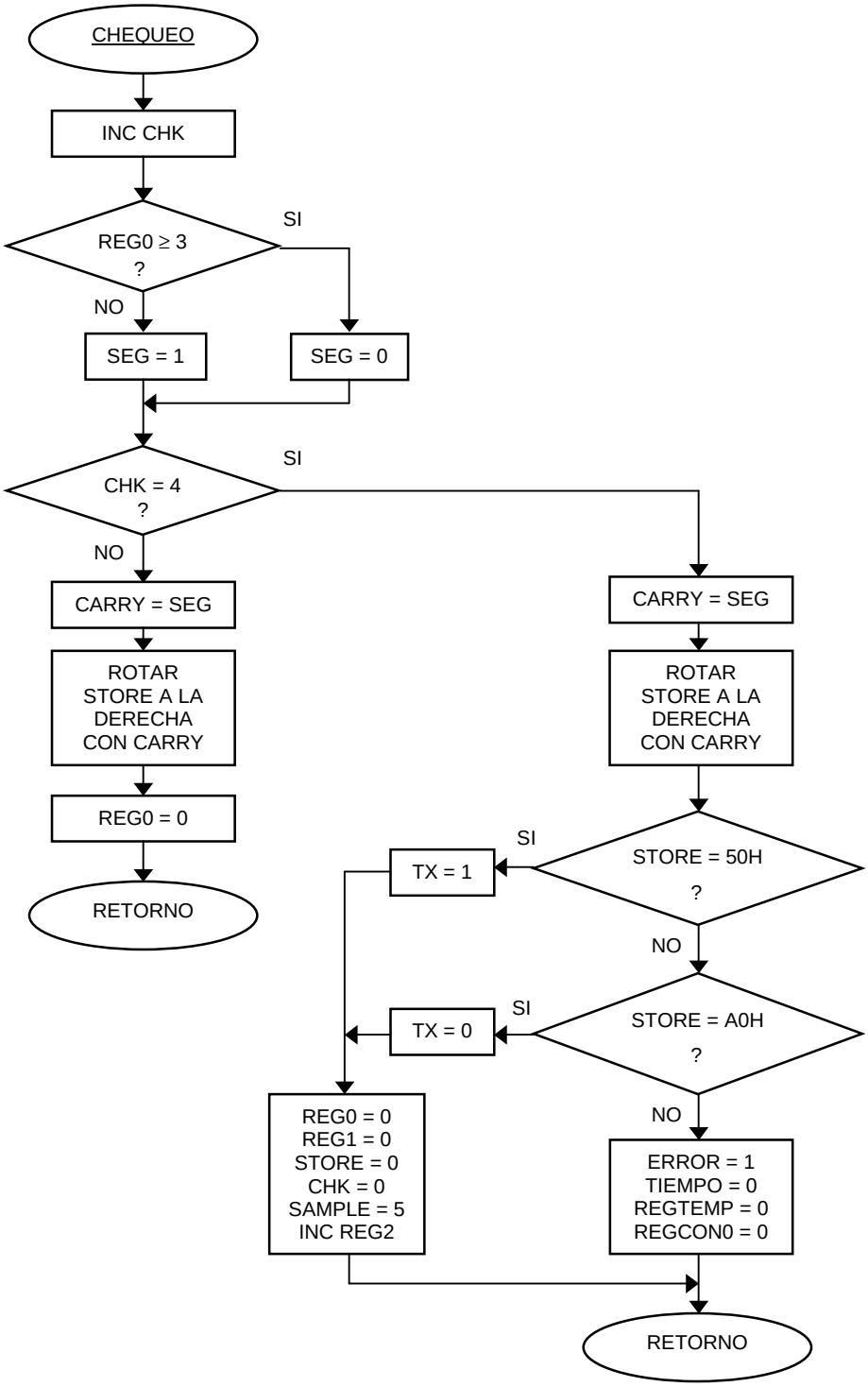
8.9.4.6 Subrutina Stop. Esta subrutina consiste en verificar que el ultimo bit leído (décimo bit) corresponda a un *Stop bit* valido, es decir, un uno lógico. La verificación se realiza revisando el contenido del bit seis (TX) del registro BUFFER, el cual contiene el estado lógico del ultimo bit demodulado. Si el estado de TX es uno indicara que se presento un *Stop bit* valido, pero si TX es cero se produce un error de comunicación, para lo cual se setea la bandera de ERROR y se clarea la bandera de TIEMPO obligando al programa a entrar de nuevo a la subrutina Tiempo de espera. En el caso de que el *Stop bit* sea valido se clarea la bandera de ERROR y se realiza el reinicio del programa, es decir, se prepara al programa para recibir un nuevo *Start bit* modulado.

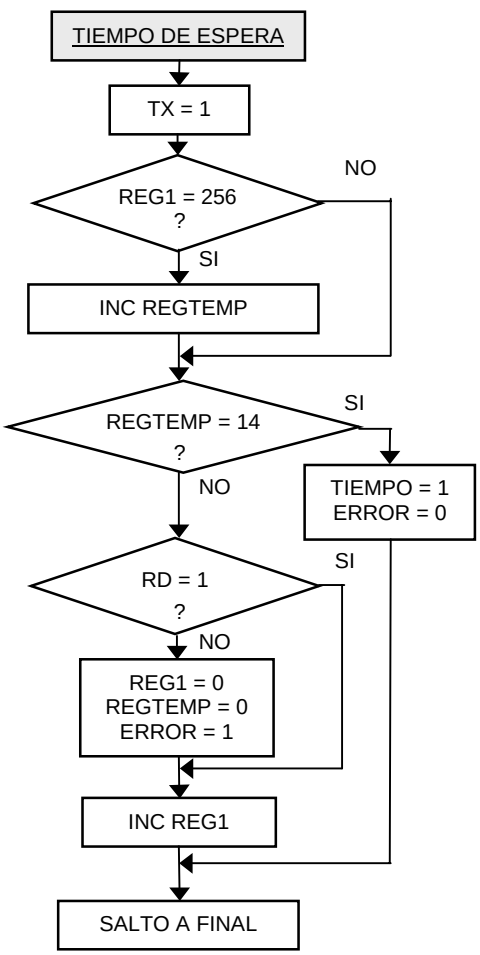
8.9.5 Diagrama de flujo del receptor. A continuación se presenta el diagrama de flujo de la rutina de servicio a la interrupción y de las subrutinas que fueron necesarias implementar en el desarrollo de la etapa de recepción para el módulo PSK.

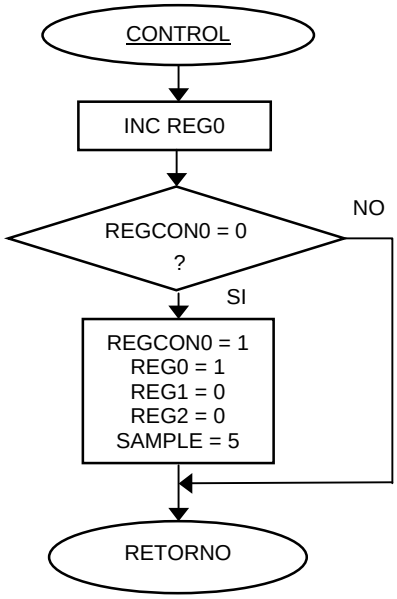
RUTINA DE SERVICIO A LA INTERRUPCION DEL RECEPTOR PSK

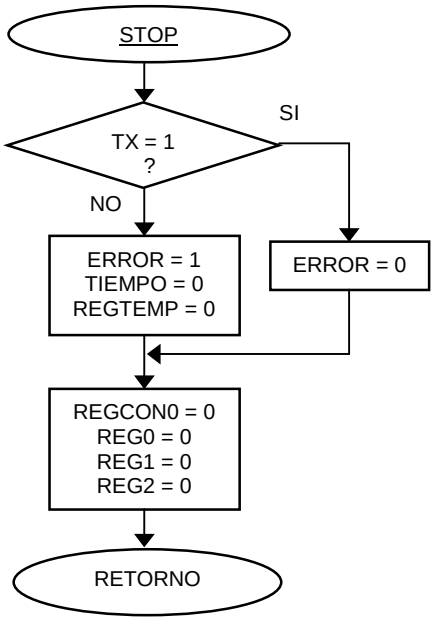


SUBRUTINAS


CHEQUEO
INC CHK
REG0 ≥ 3 ?
SI
NO
SEG = 1
SEG = 0
CHK = 4 ?
SI
NO
CARRY = SEG
ROTAR STORE A LA DERECHA CON CARRY
REG0 = 0
RETORNO
CARRY = SEG
ROTAR STORE A LA DERECHA CON CARRY
STORE = 50H ?
SI
TX = 1
NO
STORE = A0H ?
SI
TX = 0
NO
REG0 = 0
REG1 = 0
STORE = 0
CHK = 0
SAMPLE = 5
INC REG2
ERROR = 1
TIEMPO = 0
REGTEMP = 0
REGCON0 = 0
RETORNO


TIEMPO DE ESPERA
TX = 1
REG1 = 256 ?
NO
SI
INC REGTEMP
REGTEMP = 14 ?
SI
NO
TIEMPO = 1
ERROR = 0
RD = 1 ?
SI
NO
REG1 = 0
REGTEMP = 0
ERROR = 1
INC REG1
SALTO A FINAL


CONTROL
INC REG0
REGCON0 = 0 ?
NO
SI
REGCON0 = 1
REG0 = 1
REG1 = 0
REG2 = 0
SAMPLE = 5
RETORNO


STOP
TX = 1 ?
SI
NO
ERROR = 1
TIEMPO = 0
REGTEMP = 0
ERROR = 0
REGCON0 = 0
REG0 = 0
REG1 = 0
REG2 = 0
RETORNO

8.9.6 Programa en código fuente del receptor. A continuación se da el programa detallado, que se utilizo en el microcontrolador, con sus respectivos comentarios.

```
;-------------------------------------------------------------------------------------------------------------------
;                                        Demodulador PSK (Receptor)
;-------------------------------------------------------------------------------------------------------------------
;       PSKRX.ASM

        list p=16c622           ;Procesador
        include <p16c622.inc>

;       Al programar el micro tener en cuenta los fusibles de configuración
;       OSC = HS, WDT = OFF, PWRTE = OFF, CP = OFF, BODEN = OFF


;------------------------------------------------- Zona de Registros ---------------------------------------------


tmr_opt     equ     01h     ;Timer0 / option
status      equ     03h     ;
intcon      equ     0bh     ;
puertob     equ     06h     ;
buffer      equ     2ah     ;Registro que almacena cambios del puertob
reg0        equ     2bh     ;Contador de ceros
reg1        equ     2ch     ;Contador de interrupciones
reg2        equ     2dh     ;Contador de bits
regcon0     equ     2eh     ;Bandera de control de start bit
tempwr      equ     2fh     ;Temporal del reloj
store       equ     30h     ;Registro que almacena segmentos leídos
tiempo      equ     31h     ;Bandera de control del tiempo de espera
regtemp     equ     32h     ;Contador para determinar el tiempo de espera
chk         equ     33h     ;Contador de segmentos de bit leídos
sample      equ     34h     ;Contador de paquetes de dos muestras
seg         equ     35h     ;Registro que guarda el segmento actual
wtemp       equ     70h     ;Temporal de w
stemp       equ     71h     ;Temporal de status


;------------------------------------------------ Asignaciones de bit ------------------------------------------


rd          equ     7       ;Entrada serial (PSK)
tx          equ     6       ;Salida serial (RS-232)
err         equ     5       ;Aviso de error
gie         equ     7       ;Habilita int. general
toie        equ     5       ;Habilita int. por tmr0
toif        equ     2       ;Flag de int. por tmr0
rp0         equ     5       ;Página de memoria
sincro      equ     2       ;Sincronización del osciloscopio
```

```
;------------------------------------------------ Configuración del PIC ----------------------------------------------

        org     0
        goto    inicio          ;Inicio de programa principal

        org     4
        goto    inter           ;Vector de interrupción

        org     5               ;Inicio programa "inicio"

inicio  bsf     status,rp0      ;Va al banco1
        movlw   08h             ;Option, clock interno
        movwf   tmr_opt         ;
        movlw   80h             ;Configuración de puertos
        movwf   puertob         ;Puertob 7:entrada, 6-0:salida
        bcf     status,rp0      ;Retorno al banco0

        clrf    tmr_opt         ;Inicialización de registros
        clrf    puertob         ;
        clrf    reg0            ;
        clrf    reg1            ;
        clrf    reg2            ;
        clrf    regcon0         ;
        clrf    buffer          ;
        clrf    tempwr          ;
        clrf    store           ;
        clrf    tiempo          ;
        clrf    regtemp         ;
        clrf    chk             ;
        clrf    sample          ;
        clrf    seg             ;

        bsf     intcon,gie      ;Habilitación general de interrupciones
        bsf     intcon,toie     ;Habilita interrupción del timer0
        bsf     buffer,tx       ;Inicialización del puerto b
        bcf     buffer,sincro   ;

;------------------------------------------------------------ Loop ------------------------------------------------------

loop    goto    loop            ;Programa principal, loop infinito


;------------------------- RUTINA DE SERVICIO A LA INTERRUPCION DEL TMRO ----------------------


inter   movwf   wtemp           ;Push
        swapf   status,w        ;
        movwf   stemp           ;

        movf    tmr_opt,w       ;Ajuste de tiempo del timer0
        movwf   tempwr          ;
        btfss   tempwr,0        ;
        goto    time3e          ;


time3e  movlw   3eh             ;3eh -> 41,6 useg (aproximadamente)
        movwf   tmr_opt         ;Carga del timer0

princip movf    buffer,w        ;
        movwf   puertob         ;Actualización del puerto b
```

```
        btfss   tiempo,0        ;Se verifica si ya se ejecuto el tiempo de espera
        goto    time            ;

        btfss   regcon0,0       ;Mientras no se de un start bit, tx es 1 (reposo)
        goto    jump            ;

;-------------------------------------------------------- Conversión ----------------------------------------------------

        movf    sample,w        ;
        subwf   reg1,w          ;Reg1 = sample?
        btfsc   status,2        ;Si reg1 = sample -> call check
        call    check           ;Chequeo del segmento leído
        goto    leer            ;

jump    bsf     buffer,tx       ;Estado de reposo de la línea

;-------------------------------------------------------- Muestreo ------------------------------------------------------

leer    btfss   puertob,rd      ;Lectura de la entrada serial
        call    control         ;
        incf    reg1,f          ;Incrementa contador de interrupciones

stopbit movlw   0ah             ;Verificación del stop (bit 10)
        subwf   reg2,w          ;
        btfsc   status,2        ;Reg2 = 10?
        call    stop            ;Chequea si el bit 10 fue un 1
        goto    final           ;

;---------------------------------------------- Subrutina tiempo de espera ----------------------------------------------

time    bsf     buffer,tx       ;Mientras se ejecuta el tiempo de espera, tx es 1 (reposo)
        movlw   0ffh            ;Tiempo de espera de aproximadamente 150mseg
        subwf   reg1,w          ;41.6useg * 256 = 10.6mseg
        btfsc   status,2        ;
        incf    regtemp,f       ;Se cuentan ciclos de 10.6mseg

        movlw   0eh             ;
        subwf   regtemp,w       ;10.6mseg * 14 = 149.1mseg
        btfss   status,2        ;Regtemp = 14 -> tiempo = 1 y error = 0
        goto    again           ;
        bsf     tiempo,0        ;Cumplidos los 150mseg se setea la bandera de tiempo
        bcf     buffer,err      ;Se clarea la bandera de error
        goto    final           ;

again   btfsc   puertob,rd      ;Lectura de la entrada serial
        goto    unor            ;
        clrf    reg1            ;Si rd es cero durante el tiempo de espera
        clrf    regtemp         ;se inicia de nuevo el conteo
        bsf     buffer,err      ;
unor    incf    reg1,f          ;Incremento del contador de interrupciones
```

```
;---------------------------------------------------------- Final ----------------------------------------------------------

final   bcf     intcon,toif     ;Se clarea la bandera del tmr0
        swapf   stemp,w         ;Pop
        movwf   status          ;
        swapf   wtemp,f         ;
        swapf   wtemp,w         ;
        retfie                  ;

;------------------------------------------------------ Subrutina check ------------------------------------------------------


check   movlw   05h             ;Sample = sample + 5
        addwf   sample,f        ;
        incf    chk,f           ;

        movlw   03h             ;Segmento
        subwf   reg0,w          ;
        bcf     seg,0           ;Se asume seg = 0
        btfss   status,0        ;Si reg0 >= 3 -> seg = 0
        bsf     seg,0           ;

        movlw   04h             ;
        subwf   chk,w           ;Chk = 4?
        btfsc   status,2        ;Si chk = 4 -> Convierte a RS-232
        goto    equal4          ;

rotar   bcf     status,0        ;
        btfsc   seg,0           ;Carry = seg
        bsf     status,0        ;
        rrf     store,f         ;Rotación de store a la der. con carry
        clrf    reg0            ;Inicio del próximo segmento
        return                  ;Retorno

equal4  bcf     status,0        ;
        btfsc   seg,0           ;Carry = seg
        bsf     status,0        ;
        rrf     store,f         ;Ultima rotación de store a la der.
uno     movlw   50h             ;
        subwf   store,w         ;Store = 50h?
        btfss   status,2        ;Si store = 50h -> tx = 1
        goto    cero            ;
        bsf     buffer,tx       ;Conversión a RS-232 para un uno
        goto    next            ;

cero    movlw   0a0h            ;
        subwf   store,w         ;Store = a0h?
        btfss   status,2        ;Si store = a0h -> tx = 0
        goto    fail            ;
        bcf     buffer,tx       ;Conversión a RS-232 para un cero

next    incf    reg2,f          ;
        clrf    reg0            ;Iniciación de bit
        clrf    reg1            ;
        clrf    store           ;
        clrf    chk             ;
        movlw   05h             ;
        movwf   sample          ;
        return                  ;Retorno
```

```
fail    bsf     buffer,err      ;Aviso de error si el bit leído no corresponde
        clrf    tiempo          ;a un uno o un cero en PSK
        clrf    regtemp         ;
        clrf    regcon0         ;
        return                  ;Retorno

;-------------------------------------------------- Subrutina stop ----------------------------------------------------

stop    bcf     buffer,err      ;Se asume error =  0
        btfsc   buffer,tx       ;Chequeo del bit 10 para definir si hay error
        goto    valido          ;tx = 1 para el bit 10, stop valido
        bsf     buffer,err      ;tx = 0 para el bit 10, stop no valido
        clrf    tiempo          ;Se inicia de nuevo el tiempo de espera
        clrf    regtemp         ;
valido  clrf    regcon0         ;
        clrf    reg0            ;
        clrf    reg1            ;
        clrf    reg2            ;
        movlw   04h             ;Final de la trama, se permite que se vuelva a
        xorwf   buffer,f        ;sincronizar con el osciloscopio (bit 2 del puertob)
        return                  ;

;-------------------------------------------------- Subrutina control -------------------------------------------------

control incf    reg0,f          ;
        movf    regcon0,w       ;Chequea si regcon0 = 0
        btfsc   status,2        ;
        call    sinc            ;Regcon0 = 0 -> Sincronización con Start bit
        return                  ;

;-------------------------------------------------- Sincronización ----------------------------------------------------

sinc    movlw   04h             ;Señal de sincronización del osciloscopio
        xorwf   buffer,f        ;con el inicio de la trama (bit 2 del puertob)
        clrf    reg0            ;Sincronización con el start bit
        clrf    reg1            ;
        clrf    reg2            ;
        movlw   05h             ;
        movwf   sample          ;
        incf    reg0,f          ;
        bsf     regcon0,0       ;Se deshabilitan futuras sincronizaciones
        return                  ;por un 0. Se restablece con el Stop bit

;-----------------------------------------------------------------------------------------------------------------------
        end
```

8.10 MODULO ASK

Este módulo transmitirá una señal digital utilizando modulación ASK, este tipo de modulación consiste en activar o desactivar una señal portadora de acuerdo a una entrada digital.

Para la conversión a ASK primero se realiza un muestreo de la línea de entrada, esta operación es ejecutada con ayuda del Timer 0 que se encarga de generar una interrupción a intervalos de tiempo constantes (41,6μs) y durante la cual se toma una muestra del estado de la línea, que es la que contiene la información digital en forma serial.

Al acumularse un total de veinte (20) muestras, es completado un periodo de bit (832μs) y se inicia una operación de procesamiento de las mismas, es decir, se define el estado lógico que tuvo la línea, se realiza la modulación correspondiente al bit que haya sido leído y se identifica si se presentó el inicio o el final de una trama (*Start bit* o *Stop bit*), es decir la posición del bit leído dentro de la trama.

La señal de salida correspondiente al bit modulado hace también uso de la interrupción del Timer 0, ya que este también tiene la función de actualizar el estado de la línea de transmisión entre los módulos, esta actualización se hace en base al contenido de un BUFFER, el cual es un registro que almacena los cambios de estado que deben hacerse a la línea entre los módulos de transmisión y recepción.

Para generar la señal de salida del transmisor debe existir dos señales de diferente amplitud, por lo cual se conmuta una onda cuadrada (señal portadora) con una frecuencia de 4,8Khz, de esta manera un uno lógico se representa con una onda cuadrada de amplitud máxima y un cero lógico se representa por un nivel bajo de voltaje.

Para formar la onda cuadrada se cambia de estado lógico la señal de salida cada cinco interrupciones, esto se aprecia en la Figura 30.

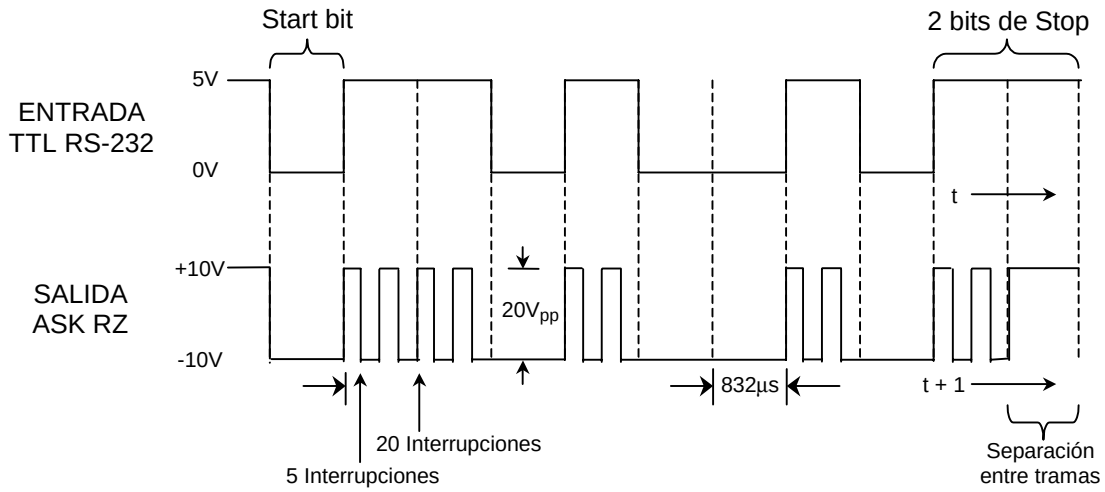


Figura 30 *Formas de onda de entrada RS-232 y salida ASK*

Se puede ver en la gráfica que un uno lógico es convertido a un tren de pulsos con una amplitud de veinte voltios pico-a-pico ($20V_{pp}$) y un cero lógico es convertido a un nivel de diez voltios negativo (-10V). Los dos voltajes de diferente polaridad son los que están presentes en la línea entre el módulo transmisor y el receptor, estos representan los niveles TTL, es decir, +10V representa un nivel alto (+5V) y -10V representa un nivel bajo (0V).

La salida ASK del módulo esta atrasada un periodo de bit (832µs), con respecto a la señal de entrada, esto se debe al tiempo de procesamiento de las veinte muestras.

La trama de datos utiliza dos bits de parada (*Stop bit*), debido a que se decidió dejar una separación entre tramas de mínimo un periodo de bit para lo cual se usa el segundo bit de parada, esto le permite al módulo detectar el bit de inicio de una próxima trama. De los dos bits sólo el primero se convierte a ASK, el segundo tendrá un estado lógico uno.

En el módulo receptor, la señal ASK es demodulada y se recupera de nuevo la trama de datos.

La demodulación se hace tomando en cuenta que uno de los estados lógicos es un tren de pulsos, el cual es detectado por medio de su frecuencia de oscilación, de esta manera el receptor se usa como conversor Frecuencia-Voltaje para determinar el bit que se debe recuperar a la salida.

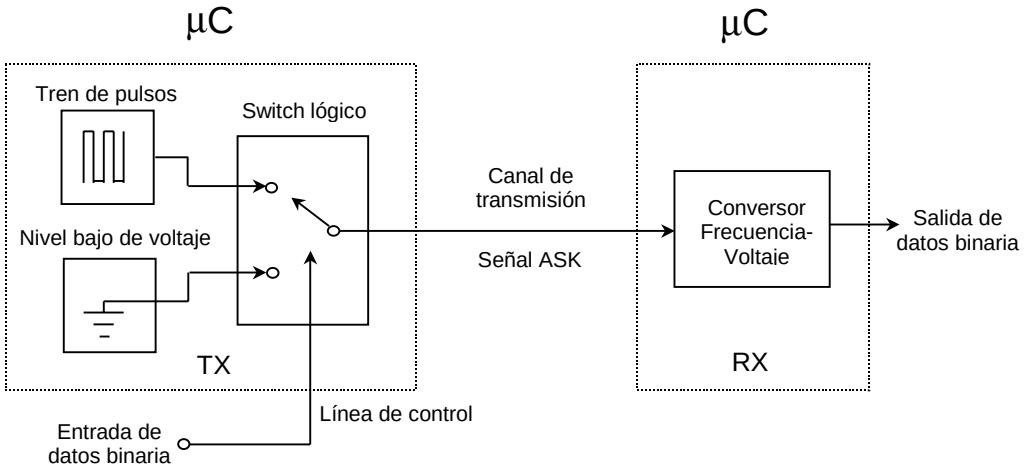


Figura 31 *Diagrama de bloques de los módulos de transmisión y recepción ASK*

La Figura 31 describe el proceso de transmisión y recepción usado por el módulo ASK. El transmisor conmuta la línea de salida entre un tren de pulsos y un nivel bajo de voltaje, la conmutación se hace de acuerdo a la entrada de datos binaria. En la etapa de recepción el conversor Frecuencia-Voltaje se encarga de recuperar los datos binarios y enviarlos a la salida.

8.10.1 Software del transmisor. El software del transmisor es quien realiza la modulación ASK de la entrada digital. La información serial de la entrada digital, la cual es entregada por un Computador, es llevada al microcontrolador por medio de un driver de línea (MAX232), este se encarga de llevar la señal a niveles de voltaje TTL, correspondientes a los niveles de voltaje que maneja el microcontrolador.

Al inicio del programa principal, se setea el bit seis del puerto B (RB6), el cual entrega la señal de salida modulada, esto se hace para indicar que la salida se encuentra en estado de reposo. Luego el programa ingresa a un loop infinito, el cual solo es interrumpido por el desbordamiento del Timer 0 que genera una interrupción. Esta interrupción dirige el programa a la dirección de memoria 0004H donde se encuentra el vector de interrupciones. En este sitio se ubica un salto que apunta a la dirección donde se encuentra la rutina de servicio a la interrupción. Esta rutina es la que finalmente se encarga del procesamiento de la entrada digital (bit siete del puerto B).

La rutina de servicio a la interrupción es la encargada de realizar el muestreo de la entrada digital y de generar la señal de salida. Al iniciarse esta rutina se entra a la subrutina Tiempo de espera, la cual realiza un retardo de aproximadamente 150ms, la línea de entrada debe de estar en estado de reposo (uno lógico) durante el tiempo de espera para continuar con el programa, el caso contrario indicara que al momento de iniciarse el programa el Computador estaba realizando una transmisión y esta será descartada. Luego del tiempo de espera el programa queda listo para recibir el *Start bit* de una trama.

Al instante en que se detecta un *Start bit* se empieza a contar veinte interrupciones (duración de un bit), durante las cuales se lee el estado lógico de la entrada, al completarse estas veinte interrupciones se entra a la subrutina Txout donde se verifica el estado lógico del bit leído y se decide la amplitud del bit modulado. Este proceso de conteo de veinte interrupciones se repite con los ocho bits de datos y el *Stop bit*. Posterior a la verificación hecha por la subrutina Txout y de acuerdo a la amplitud seleccionada se realiza un conteo de interrupciones durante los cuales se harán cambios de estado para formar la señal de onda cuadrada.

Al terminar el muestreo y la modulación de la trama completa se verifica que el ultimo bit leído sea un *Stop bit* (uno lógico), si es así se realiza un reinicio del programa, es decir, se vuelve a las condiciones de espera de un próximo *Start bit*, pero en el caso en que el ultimo bit leído sea un cero lógico, significara que

hay un error de transmisión, por lo cual se dará un aviso de error y se entrara de nuevo a la subrutina Tiempo de espera.

A continuación se hace una descripción detallada de las rutinas implementadas para el desarrollo del modulador ASK.

8.10.1.1 Programa principal. En esta parte del programa se configura el Timer 0, el cual utiliza la fuente de reloj interna del microcontrolador, se programan el bit siete del puerto B como entrada y los bits cero a seis del puerto B como salida, se inicializan todos los registros que se utilizan en el programa, se habilita la interrupción del Timer 0 y se setea el bit seis del puerto B para indicar que la salida se encuentra en estado de reposo. Esto es realizado únicamente al inicio del programa y no se vuelve a entrar en el mientras el programa este corriendo. Luego el programa ingresa a un loop infinito, es decir, se ejecuta un salto que apunta a la dirección de memoria donde se encuentra el mismo salto, este lazo cerrado que se produce solo se interrumpe por la acción de desbordamiento del Timer 0 cada 41,6µs para procesar la entrada digital.

8.10.1.2 Rutina de servicio a la interrupción del Timer 0. Esta rutina inicialmente ejecuta un PUSH, el cual se encarga de almacenar el registro de trabajo W y las banderas de estado de la ALU como Carry, Digit Carry y Cero que se encuentran en el registro STATUS. Luego se realiza el ajuste de tiempo

del Timer 0, este ajuste se realiza para cargar el Timer 0 correctamente y de esta manera calibrar el tiempo en que se presentara la próxima interrupción.

A continuación se realiza una actualización del puerto B que consiste en mandar a las salidas del puerto los cambios que se hayan hecho al registro BUFFER. Después se verifica que la bandera TIEMPO sea uno, esto indica que el tiempo de espera ya ha sido ejecutado, en el caso en que sea cero se salta a la subrutina Tiempo de espera. El registro REGCON0 es el registro encargado de controlar la sincronización del programa con el *Start bit* de una trama, si este registro es uno indicara que ya se presento un *Start bit* por lo tanto se debe empezar el conteo de las veinte interrupciones, este conteo lo realiza el registro REG1.

Al completarse veinte interrupciones se llama a la subrutina Txout, esta se encarga de verificar el estado lógico del bit leído y decidir la amplitud que deberá tener la señal modulada de salida. El registro REG2 es quien lleva el conteo del numero de bits leídos, si este es diferente de cero indicara que ya se ha leído el primer bit, por lo tanto se debe empezar la modulación, para realizarla se verifica el registro SQUARE, este actúa como bandera para indicar que se debe modular un uno lógico, si este es uno se genera una onda cuadrada de 4,8KHz, para lo cual se efectúan cinco cambios de estado durante veinte interrupciones, en el caso de que el registro SQUARE sea cero, indicara que se modulara un cero, para lo cual no se genera la onda cuadrada. Cada cambio de estado se envía al bit seis del registro BUFFER.

Al realizar la lectura del bit siete (RD) del puerto B, se verifica el estado lógico de la entrada digital, si es un cero lógico se llama a la subrutina Control, la cual se encarga de llevar el conteo del numero de ceros lógicos que se han leído y de controlar la sincronización del programa con el *Start bit*.

Si el registro REG2 es igual a diez indicara que ya se ha leído la trama completa, por lo que se llama a la subrutina Stop, que se encarga de verificar que el ultimo bit leído sea un *Stop bit* valido (uno lógico). Al haberse completado la lectura de la trama completa se produce un reinicio del programa, es decir, se prepara al programa para detectar un próximo *Start bit*, pero este reinicio se ejecuta quince interrupciones después de que se ha leído el *Stop bit*, debido a que se debe permitir que este ultimo termine de ser modulado, ya que la modulación se hace después de realizar la lectura.

Por ultimo se ejecuta un POP para restaurar el registro W y las banderas de Carry, Digit Carry y Cero que se habían almacenado a su registro original (STATUS).

8.10.1.3 Subrutina Tiempo de espera. La subrutina de tiempo de espera consiste en un retardo de 150ms, esto se logra contando las interrupciones del Timer 0 que se producen cada 41,6µs, de esta manera se realiza el conteo de 256 interrupciones lo que proporciona un tiempo de 10,7ms, para lograr un tiempo de 150ms se cuentan 17 ciclos de 10,7ms.

Mientras se ejecuta el tiempo de espera se realiza una lectura de la entrada digital, si el resultado de la lectura es un uno lógico (estado de reposo) se sigue con el conteo hasta completar el tiempo de espera requerido, después de cumplido esto el registro TIEMPO, el cual actúa como bandera para informar que el tiempo de espera ha sido ejecutado, se carga con un uno y la bandera de ERROR (bit cinco del puerto B) se clarea, pero si la lectura da como resultado un cero lógico indicara que una transmisión ha comenzado antes de dar inicio al programa y esto provocara un error de transmisión, lo cual hace que se reinicie de nuevo el tiempo de espera y se setea la bandera de ERROR.

8.10.1.4 Subrutina Txout. Esta subrutina revisa el registro REG0, el cual contiene el numero de lecturas que correspondieron a un cero lógico durante las veinte interrupciones, si de las veinte lecturas realizadas once o más fueron un cero lógico indicara que el bit leído es un cero, pero si de las veinte lecturas tomadas diez o menos fueron un cero lógico quiere decir que el bit leído es un uno. De esta manera se realiza un promedio de las lecturas para descartar los posibles cambios en la señal debidos al ruido que se pueda presentar.

Si el bit leído es un cero, la señal de salida deberá tener una amplitud de cero voltios, por lo cual la señal modulada tendrá un estado inicial bajo (cero lógico), pero si el bit leído es un uno, la señal de salida deberá ser una onda cuadrada de amplitud máxima (cinco voltios), por lo cual la señal modulada tendrá un estado inicial alto (uno lógico) y para indicar al programa que se debe generar una onda cuadrada se utiliza el registro SQUARE, el cual se carga con un uno

para esta finalidad. El estado inicial de la señal modulada es puesto en el bit seis (TX) del registro BUFFER. El resultado de las veinte lecturas, es decir, uno o cero lógico, es almacenado en el registro REGAUX. Luego se prepara al programa para leer el próximo bit.

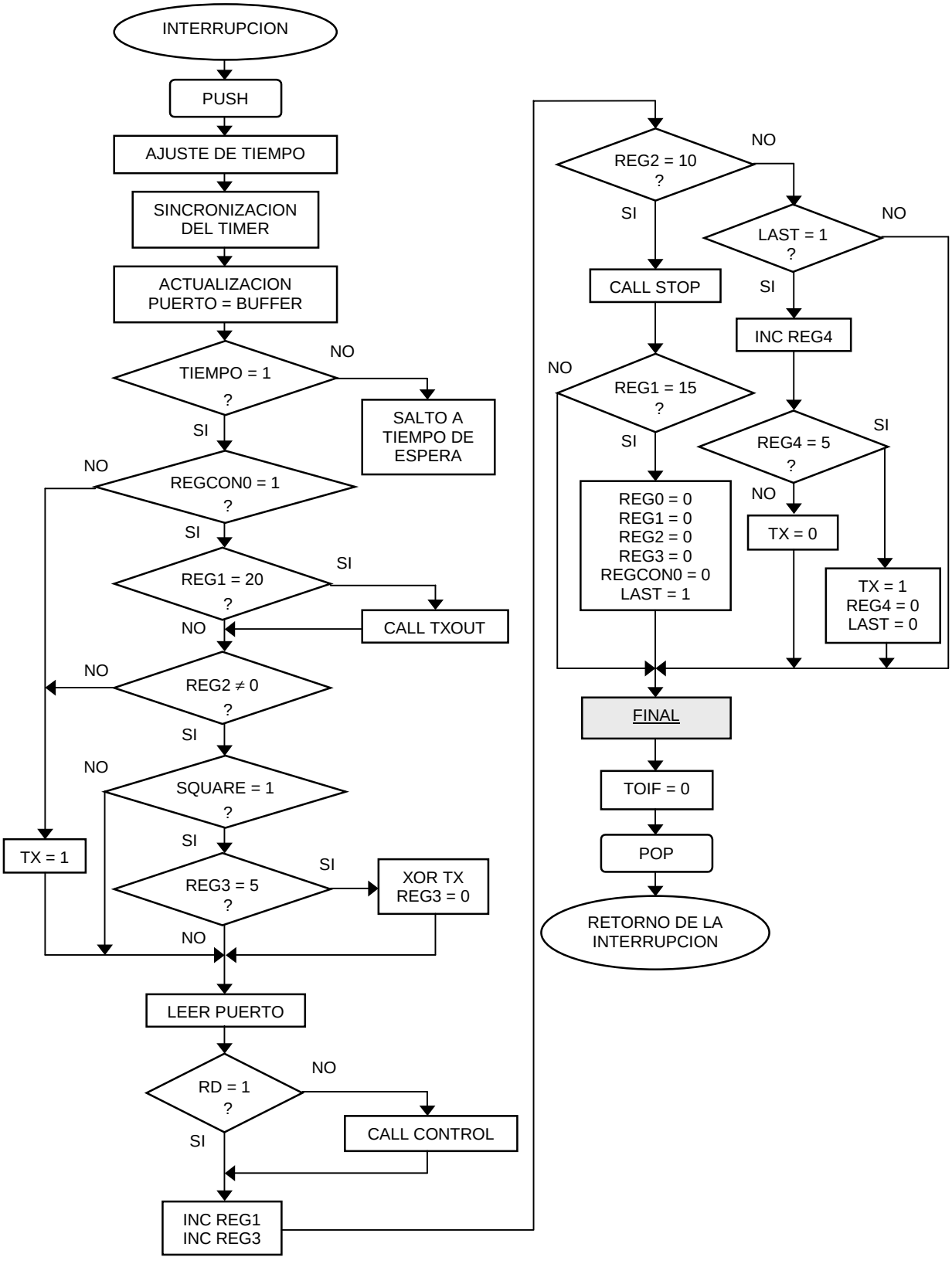
8.10.1.5 Subrutina Control. Esta subrutina es quien realiza el conteo de las lecturas que corresponden a un cero lógico, incrementando el registro REG0. También realiza la sincronización del programa con el *Start bit* de una trama, esta sincronización consiste en preparar al programa para empezar la lectura de todos los bits de la trama, esto se logra inicializando los registros encargados de este proceso. La subrutina Control utiliza el registro REGCON0 como bandera de control para indicar que la sincronización ya se efectúo, la sincronización solo se hace cuando se detecta el *Start bit* y no se realiza mientras se lee la trama completa, después de que esta ha sido leída y modulada el programa queda a la espera de un nuevo *Start bit* y de una nueva sincronización.

8.10.1.6 Subrutina Stop. Esta subrutina consiste en verificar si el ultimo bit leído (décimo bit) corresponde a un *Stop bit* valido, es decir, un uno lógico. La verificación se realiza revisando el contenido del registro REGAUX, el cual almacena el ultimo bit leído. Si el registro REGAUX es un uno, indicara que se leyó un *Stop bit* que es valido, de esta manera se clarea la bandera de ERROR y se retorna, pero en el caso contrario, es decir, si el registro REGAUX es cero

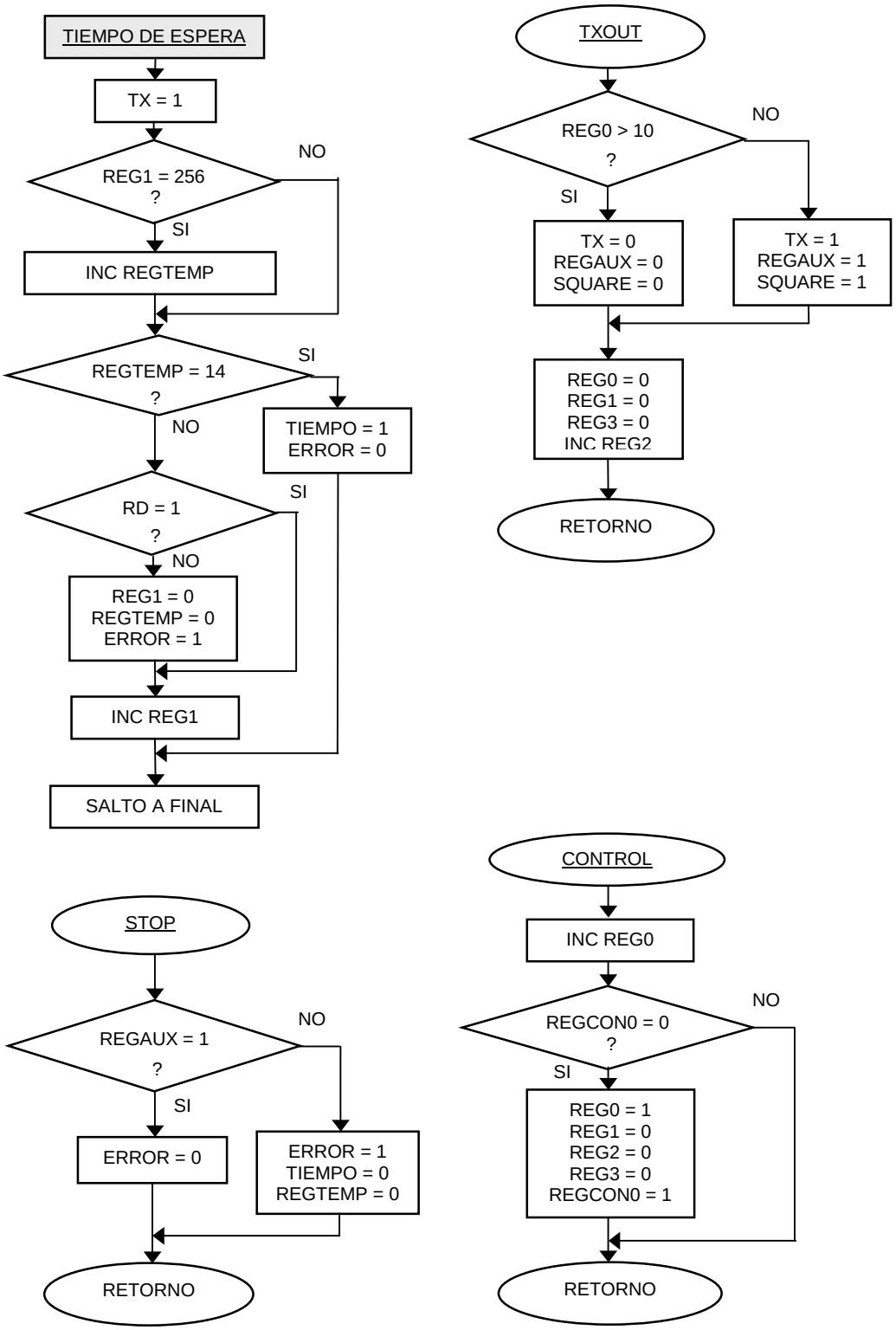
indica que el *Stop bit* leído no es valido causando un error de comunicación, para lo cual se setea la bandera de ERROR y se clarea la bandera de TIEMPO obligando al programa a entrar de nuevo a la subrutina Tiempo de espera.

8.10.2 Diagrama de flujo del transmisor. A continuación se presenta el diagrama de flujo de la rutina de servicio a la interrupción y de las subrutinas que fueron necesarias implementar en el desarrollo de la etapa de transmisión para el módulo ASK.

RUTINA DE SERVICIO A LA INTERRUPCION DEL TRANSMISOR ASK



SUBRUTINAS


TIEMPO DE ESPERA
TX = 1
REG1 = 256 ?
NO
SI
INC REGTEMP
REGTEMP = 14 ?
SI
NO
TIEMPO = 1
ERROR = 0
RD = 1 ?
SI
NO
REG1 = 0
REGTEMP = 0
ERROR = 1
INC REG1
SALTO A FINAL
TXOUT
REG0 > 10 ?
NO
SI
TX = 0
REGAUX = 0
SQUARE = 0
TX = 1
REGAUX = 1
SQUARE = 1
REG0 = 0
REG1 = 0
REG3 = 0
INC REG2
RETORNO
STOP
REGAUX = 1 ?
NO
SI
ERROR = 0
ERROR = 1
TIEMPO = 0
REGTEMP = 0
RETORNO
CONTROL
INC REG0
REGCON0 = 0 ?
NO
SI
REG0 = 1
REG1 = 0
REG2 = 0
REG3 = 0
REGCON0 = 1
RETORNO

8.10.3 Programa en código fuente del transmisor. A continuación se da el programa detallado, que se utilizo en el microcontrolador, con sus respectivos comentarios.

```
;-------------------------------------------------------------------------------------------------------------------
;                                        Modulador ASK (Transmisor)
;-------------------------------------------------------------------------------------------------------------------
;       ASKTX.ASM

        list p=16c622           ;Procesador
        include <p16c622.inc>

;       Al programar el micro tener en cuenta los fusibles de configuración
;       OSC = HS, WDT = OFF, PWRTE = OFF, CP = OFF, BODEN = OFF


;---------------------------------------------------- Zona de Registros ----------------------------------------------


tmr_opt     equ     01h     ;Timer0 / option
status      equ     03h     ;
intcon      equ     0bh     ;
puertob     equ     06h     ;
buffer      equ     2ah     ;Registro que almacena cambios del puertob
reg0        equ     2bh     ;Contador de ceros
reg1        equ     2ch     ;Contador de interrupciones
reg2        equ     2dh     ;Contador de bits
reg3        equ     2eh     ;Contador de paquetes de 5 interrupciones
reg4        equ     2fh     ;Contador de las ultimas 5 interrupciones de una trama
reg5        equ     30h     ;Contador de interrupciones para los osciladores
regaux      equ     31h     ;Registro auxiliar que almacena el contenido de tx
regcon0     equ     32h     ;Bandera de control de start bit
tempwr      equ     34h     ;Temporal de reloj
tiempo      equ     35h     ;Bandera de control del tiempo de espera
regtemp     equ     36h     ;Contador para determinar el tiempo de espera
square      equ     37h     ;Bandera para habilitación de onda cuadrada
last        equ     38h     ;Bandera para habilitar tx en las 5 int. finales
wtemp       equ     70h     ;Temporal de w
stemp       equ     71h     ;Temporal de status


;------------------------------------------------- Asignaciones de bit -----------------------------------------------


rd          equ     7       ;Entrada serial (RS-232)
tx          equ     6       ;Salida serial (ASK)
err         equ     5       ;Aviso de error
gie         equ     7       ;Habilita int general
toie        equ     5       ;Habilita int por tmr0
toif        equ     2       ;Flag de int por tmr0
rp0         equ     5       ;Página de memoria
osc1        equ     4       ;Portadora
osc2        equ     3       ;Portadora desplazada
sincro      equ     2       ;Sincronización del osciloscopio
```

```
;------------------------------------------------ Configuración del PIC ------------------------------------------

        org     0
        goto    inicio          ;Inicio de programa principal

        org     4
        goto    inter           ;Vector de interrupción

        org     5               ;Inicio programa "inicio"

inicio  bsf     status,rp0      ;Va al banco1
        movlw   08h             ;Option, clock interno
        movwf   tmr_opt         ;
        movlw   80h             ;Configuración de puertos
        movwf   puertob         ;Puertob 7:entrada, 6-0:salida
        bcf     status,rp0      ;Retorno banco0

        clrf    tmr_opt         ;Inicialización de registros
        clrf    puertob         ;
        clrf    reg0            ;
        clrf    reg1            ;
        clrf    reg2            ;
        clrf    reg3            ;
        clrf    reg4            ;
        clrf    reg5            ;
        clrf    regaux          ;
        clrf    regcon0         ;
        clrf    buffer          ;
        clrf    tempwr          ;
        clrf    tiempo          ;
        clrf    regtemp         ;
        clrf    square          ;
        clrf    last            ;

        bsf     intcon,gie      ;Habilitación general de interrupciones
        bsf     intcon,toie     ;Habilita interrupción de timer0
        bsf     buffer,tx       ;Inicialización del puerto b
        bsf     buffer,osc1     ;Inicialización de señales generadas
        bcf     buffer,osc2     ;
        bcf     buffer,sincro   ;

;------------------------------------------------------------ Loop ------------------------------------------------

loop    goto    loop            ;Programa principal, Loop infinito


;------------------------ RUTINA DE SERVICIO A LA INTERRUPCION DEL TMRO ----------------------

inter   movwf   wtemp           ;Push
        swapf   status,w        ;
        movwf   stemp           ;

        movf    tmr_opt,w       ;Ajuste de tiempo del timer0
        movwf   tempwr          ;
        btfss   tempwr,0        ;
        goto    time3e          ;

time3e  movlw   3eh             ;3eh -> 41,6 useg (aproximadamente)
        movwf   tmr_opt         ;Carga del timer0
```

```
princip  movf    buffer,w        ;
         movwf   puertob         ;Actualización del puerto b

         btfss   tiempo,0        ;Se verifica si ya se ejecuto el tiempo de espera
         goto    time            ;

         btfss   regcon0,0       ;Mientras no se de un start bit, Tx es 1 (reposo)
         goto    jump            ;

;---------------------------------------------------------- Conversión ----------------------------------------------------------


         movlw   14h             ;
         subwf   reg1,w          ;Reg1 = 20?
         btfsc   status,2        ;Si reg1 = 20 -> call txout
         call    txout           ;Actualización de tx

         movf    reg2,w          ;Si reg2 es diferente de cero se permite
         btfsc   status,2        ;la conversión del primer bit leído
         goto    jump

         btfss   square,0        ;Si Square es uno sale una onda cuadrada
         goto    leer            ;

         movlw   05h             ;Se cuentan 5 interrupciones
         subwf   reg3,w          ;
         btfss   status,2        ;
         goto    leer            ;

         movlw   40h             ;XOR de tx cada 5 interrupciones
         xorwf   buffer,f        ;
         clrf    reg3            ;
         goto    leer            ;

jump     bsf     buffer,tx       ;Estado de reposo de la línea

;---------------------------------------------------------- Muestreo ----------------------------------------------------------


leer     btfss   puertob,rd      ;Lectura de la entrada serial
         call    control         ;
         incf    reg1,f          ;Incrementa contador de interrupciones
         incf    reg3,f          ;Incrementa contador de paquetes de 5 interrupciones
         incf    reg5,f          ;Incrementa contador de osciladores

;---------------------------------------------------------- Stop ----------------------------------------------------------


stopbit  movlw   0ah             ;Verificación del stop (bit 10)
         subwf   reg2,w          ;
         btfss   status,2        ;Reg2 = 10?
         goto    ultimo          ;
         bcf     buffer,err      ;Se asume error = 0
         btfsc   regaux,0        ;Chequeo del bit 10 para definir si hay error
         goto    valido          ;Tx = 1 para el bit 10, stop valido
         bsf     buffer,err      ;Tx = 0 para el bit 10, stop no valido
         clrf    tiempo          ;Se inicia de nuevo el tiempo de espera
         clrf    regtemp         ;
         goto    final           ;
```

```
;---------------------------------------------------------- Reset ----------------------------------------------------------


valido  movlw   0fh             ;El Reset se da en la interrupción 15 después
        subwf   reg1,w          ;de que se ha leído el stop bit
        btfss   status,2        ;
        goto    final           ;
        clrf    regcon0         ;Reset
        clrf    reg0            ;
        clrf    reg1            ;
        clrf    reg2            ;
        clrf    reg3            ;
        bsf     last,0          ;Habilita la salida de Tx en las 5 int. finales
        goto    final           ;


;---------------------------------------------------------- Last ----------------------------------------------------------


ultimo  btfss   last,0          ;
        goto    final           ;

        incf    reg4,f          ;Conteo de las 5 interrupciones faltantes
        movlw   05h             ;
        subwf   reg4,w          ;
        btfss   status,2        ;
        goto    recover         ;

        clrf    last            ;Finalización de la recuperación del 4 segmento
        clrf    reg4            ;
        movlw   04h             ;Final de la trama, se permite que se vuelva a
        xorwf   buffer,f        ;sincronizar con el osciloscopio (bit 2 del puertob)
        bsf     buffer,tx       ;
        goto    final           ;

recover bcf     buffer,tx       ;Recuperación del cuarto segmento
        goto    final           ;


;------------------------------------------- Subrutina tiempo de espera -------------------------------------------


time    bsf     buffer,tx       ;Mientras se ejecuta el tiempo de espera, tx es 1 (reposo)
        movlw   0ffh            ;Tiempo de espera de aproximadamente 150mseg
        subwf   reg1,w          ;41.6useg * 256 = 10.6mseg
        btfsc   status,2        ;
        incf    regtemp,f       ;Se cuentan ciclos de 10.6mseg
        movlw   0eh             ;
        subwf   regtemp,w       ;10.6mseg * 14 = 149.1mseg
        btfss   status,2        ;Regtemp = 14 -> tiempo = 1 y error = 0
        goto    again           ;
        bsf     tiempo,0        ;Cumplidos los 150mseg se setea la bandera de tiempo
        bcf     buffer,err      ;Se clarea la bandera de error
        goto    final           ;

again   btfsc   puertob,rd      ;Lectura de la entrada serial
        goto    uno             ;
        clrf    reg1            ;Si rd es cero durante el tiempo de espera
        clrf    regtemp         ;se inicia de nuevo el conteo
        bsf     buffer,err      ;
uno     incf    reg1,f          ;Incremento de contador de interrupciones
```

```
;---------------------------------------------------------- Final ----------------------------------------------------------

final   movlw   05h             ;Generación de la portadora
        subwf   reg5,w          ;(bit 4 del puertob)
        btfss   status,2        ;
        goto    salir           ;
        movlw   10h             ;
        xorwf   buffer,f        ;
        clrf    reg5            ;

salir   bcf     intcon,toif     ;Se clarea la bandera del tmr0
        swapf   stemp,w         ;Pop
        movwf   status          ;
        swapf   wtemp,f         ;
        swapf   wtemp,w         ;
        retfie                  ;

;------------------------------------------------------ Subrutina txout ----------------------------------------------------

txout   movf    reg0,w          ;
        sublw   0ah             ;(10 - reg0)
        btfss   status,0        ;Si reg0 > 10 -> tx=0
        goto    tx0             ;
        bsf     buffer,tx       ;tx=1
        bsf     square,0        ;
        bsf     regaux,0        ;Se almacena bit leído en Regaux
        goto    done            ;
tx0     bcf     buffer,tx       ;tx=0
        bcf     square,0        ;
        bcf     regaux,0        ;Se almacena bit leído en Regaux
done    incf    reg2,f          ;
        clrf    reg0            ;Iniciación de bit
        clrf    reg1            ;
        clrf    reg3            ;
        return                  ;

;------------------------------------------------------ Subrutina control --------------------------------------------------

control incf    reg0,f          ;
        movf    regcon0,w       ;Chequea si regcon0 = 0
        btfsc   status,2        ;
        call    sinc            ;Regcon0 = 0
        return                  ;

;------------------------------------------------------ Sincronización -----------------------------------------------------

sinc    movlw   04h             ;Señal de sincronización del osciloscopio
        xorwf   buffer,f        ;con el inicio de la trama (bit 2 del puertob)
        clrf    reg0            ;Sincronización con el start bit
        clrf    reg1            ;
        clrf    reg2            ;
        clrf    reg3            ;
        incf    reg0,f          ;
        bsf     regcon0,0       ;Se deshabilitan futuras sincronizaciones
        return                  ;por un 0. Se restablece con el Stop bit

;-------------------------------------------------------------------------------------------------------------------------
        end
```

8.10.4 Software del receptor. El software del receptor es el encargado de demodular la señal ASK y recuperar la información digital. Este software divide el bit modulado en cuatro segmentos, cada segmento esta compuesto por cinco interrupciones, durante las cuales se realiza la lectura de la entrada modulada, al final de estas cinco interrupciones se define el estado lógico del segmento leído y se almacena en el registro STORE. Al terminar de leer el bit modulado, en el registro STORE quedara guardada la secuencia de segmentos y esta ayudara a determinar la amplitud del bit modulado.

Al inicio del programa principal se setea el bit seis del puerto B (RB6), el cual entrega la señal de salida demodulada, esto se hace para indicar que la salida se encuentra en estado de reposo. Luego el programa ingresa a un loop infinito, el cual solo es interrumpido por el desbordamiento del Timer 0 que genera una interrupción. Esta interrupción dirige el programa a la dirección de memoria 0004H donde se encuentra el vector de interrupciones. En este sitio se ubica un salto que apunta a la dirección donde se encuentra la rutina de servicio a la interrupción. Es esta rutina la que finalmente se encarga del procesamiento de la entrada modulada (bit siete del puerto B).

La rutina de servicio a la interrupción es la encargada de realizar el muestreo de la entrada modulada y de generar la señal de salida. Al iniciarse esta rutina se entra a la subrutina Tiempo de espera, la cual realiza un retardo de aproximadamente 150ms en el cual se verifica que el Computador no este transmitiendo al momento de iniciar el programa.

Luego de ejecutar el tiempo de espera el programa queda listo para recibir el *Start bit* modulado de una trama. Al instante en que se detecta un *Start bit* se empieza a contar cinco interrupciones, durante las cuales se lee el estado lógico de la entrada, al completarse estas cinco interrupciones (duración de un segmento del bit modulado) se entra a la subrutina Chequeo donde se verifica el estado lógico del segmento leído y se almacena, este proceso es repetido hasta obtener los cuatro segmentos correspondientes a un bit modulado, lo cual tarda veinte interrupciones, al final de las cuales se determina la amplitud del bit modulado y se define el estado lógico de la salida digital. Los ocho bits de datos y el *Stop bit* son demodulados de la misma forma.

Al terminar el muestreo y la demodulación de la trama completa, se verifica que el ultimo bit leído sea un *Stop bit* valido, si es así se realiza un reinicio del programa, es decir, se vuelve a las condiciones de espera de un próximo *Start bit*, pero en el caso en que el ultimo bit leído sea un cero lógico, significara que hay un error de transmisión, por lo cual se dará un aviso de error y se entrara de nuevo a la subrutina Tiempo de espera.

A continuación se hace una descripción detallada de las rutinas implementadas para el desarrollo del demodulador ASK.

8.10.4.1 Programa principal. En esta parte del programa se configura el Timer 0, el cual utiliza la fuente de reloj interna del microcontrolador, se programan el bit siete del puerto B como entrada y los bits cero a seis del

puerto B como salida, se inicializan todos los registros que se utilizan en el programa, se habilita la interrupción del Timer 0 y se setea el bit seis del puerto B para indicar que la salida se encuentra en estado de reposo. Esto es realizado únicamente al inicio del programa y no se vuelve a entrar en el mientras el programa este corriendo. Luego el programa ingresa a un loop infinito, es decir, se ejecuta un salto que apunta a la dirección de memoria donde se encuentra el mismo salto, este lazo cerrado que se produce solo se interrumpe por la acción de desbordamiento del Timer 0 cada 41,6μs para procesar la entrada modulada.

8.10.4.2 Rutina de servicio a la interrupción del Timer 0. Esta rutina inicialmente ejecuta un PUSH, el cual se encarga de almacenar el registro de trabajo W y las banderas de estado de la ALU. Luego se realiza el ajuste de tiempo del Timer 0, este ajuste se realiza para cargar el Timer 0 correctamente y de esta manera calibrar el tiempo en que se presentara la próxima interrupción.

Posteriormente se realiza una actualización del puerto B que consiste en mandar a las salidas del puerto los cambios que se hayan hecho al registro BUFFER. Después se verifica que la bandera TIEMPO sea uno, esto indica que el tiempo de espera ya ha sido ejecutado, en el caso en que sea cero se salta a la subrutina Tiempo de espera. El registro REGCON0 es el registro encargado de controlar la sincronización del programa con el *Start bit* modulado, si este registro es uno indicara que ya se presento un *Start bit* por lo

tanto se debe empezar el conteo de interrupciones, este conteo lo realiza el registro REG1.

A intervalos de cinco interrupciones se llama a la subrutina Chequeo, esta se encarga de verificar y almacenar el estado lógico del segmento leído, esto se repite hasta completar cuatro segmentos, lo cual toma veinte interrupciones, al completar cuatro segmentos se determina la fase del bit modulado y se define el estado lógico de la salida digital. Este proceso se repite con los ocho bits de datos y el *Stop bit*.

Al realizar la lectura del bit siete (RD) del puerto B, se verifica el estado lógico de la entrada modulada, si es un cero lógico se llama a la subrutina Control, la cual se encarga de llevar el conteo del numero de ceros lógicos que se han leído y de controlar la sincronización del programa con el *Start bit* modulado.

Si el registro REG2 es igual a diez, indicara que se ha leído la trama completa, por lo que se llama a la subrutina Stop, que se encarga de verificar que el ultimo bit leído sea un *Stop bit* valido. Dentro de esta subrutina se produce el reinicio del programa, es decir, se prepara al programa para detectar un próximo *Start bit*.

Por ultimo se ejecuta un POP para restaurar el registro W y las banderas de estado de la ALU a su registro original (STATUS).

8.10.4.3 Subrutina Tiempo de espera. La subrutina de tiempo de espera consiste en un retardo de 150ms, esto se logra contando las interrupciones del Timer 0 que se producen cada 41,6μs, de esta manera se realiza el conteo de 256 interrupciones lo que proporciona un tiempo de 10,7ms, para lograr un tiempo de 150ms se cuentan 14 ciclos de 10,7ms.

Mientras se ejecuta el tiempo de espera se realiza una lectura de la entrada modulada, si el resultado de la lectura es un uno lógico (estado de reposo) se sigue con el conteo hasta completar el tiempo de espera requerido, después de cumplido esto el registro TIEMPO, el cual actúa como bandera para informar que el tiempo de espera ha sido ejecutado, se carga con un uno y la bandera de ERROR (bit cinco del puerto B) se clarea, pero si la lectura da como resultado un cero lógico indicara que una transmisión ha comenzado antes de dar inicio al programa y esto provocara un error de transmisión, lo cual hace que se reinicie de nuevo el tiempo de espera y se setea la bandera de ERROR.

8.10.4.4 Subrutina Chequeo. Esta subrutina revisa el registro REG0 a intervalos de cinco interrupciones, el REG0 contiene el numero de lecturas que correspondieron a un cero lógico durante cinco interrupciones, si de las cinco lecturas realizadas tres o más fueron un cero lógico esto indicara que el segmento leído es un cero, pero si dos o menos de las lecturas tomadas fueron un cero lógico quiere decir que el segmento leído es un uno. De esta manera se realiza un promedio de las lecturas para descartar los posibles cambios en la señal debidos al ruido que se pueda presentar. Se acumulan entonces cuatro

segmentos durante veinte interrupciones, cada uno de los segmentos es almacenado en el registro STORE.

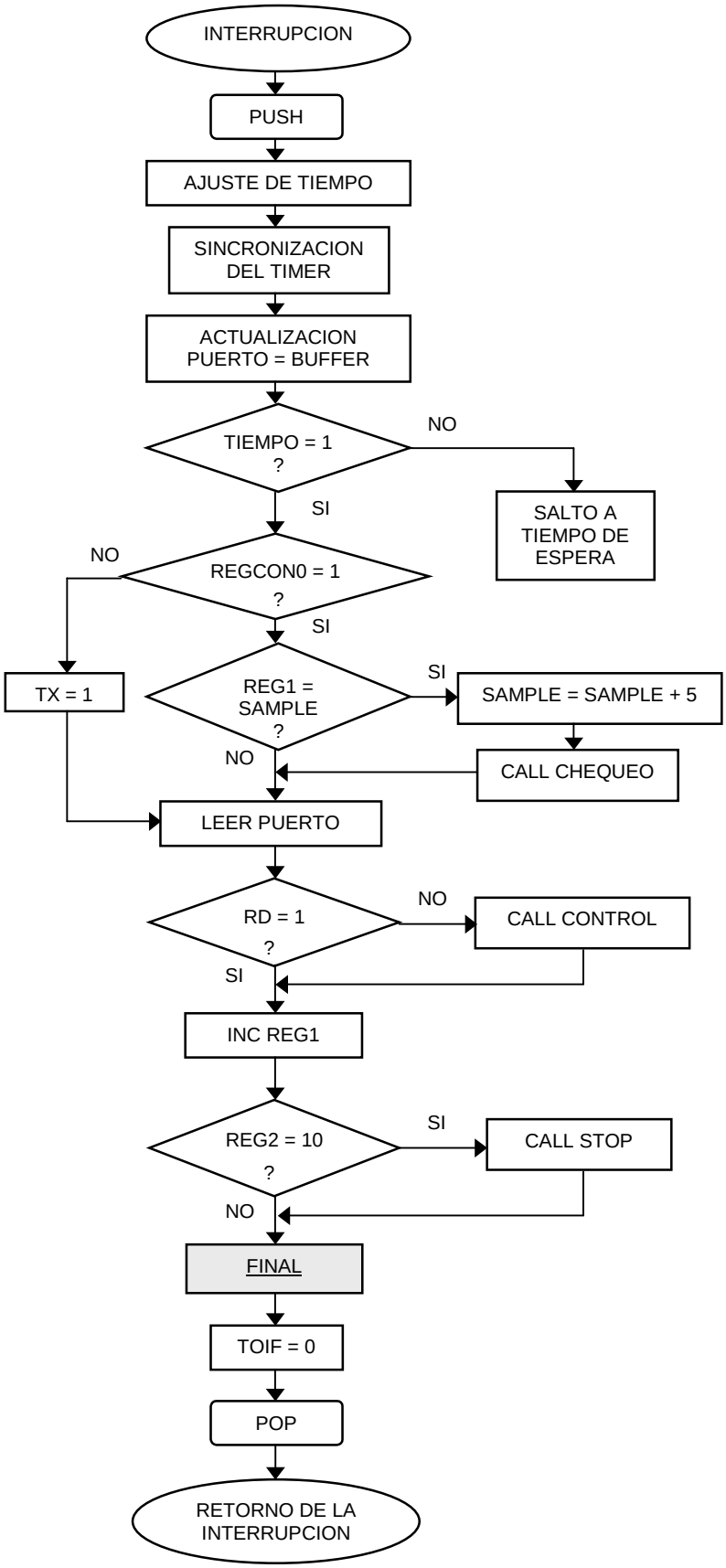
Al completar cuatro segmentos, en el registro STORE queda una secuencia de unos y ceros que ayuda a determinar la amplitud del bit modulado. Si el registro STORE es igual a 50H indicara que el bit modulado es una onda cuadrada con una amplitud de cinco voltios, por lo tanto la salida digital, por la cual se obtiene el bit demodulado, deberá tener un estado lógico uno, pero si el registro STORE es igual a 00H indicara que la amplitud del bit modulado es cero voltios y la salida digital deberá tener un estado lógico cero. Luego se prepara al programa para leer el próximo bit.

8.10.4.5 Subrutina Control. Esta subrutina es quien realiza el conteo de las lecturas que corresponden a un cero lógico, incrementando el registro REG0. También realiza la sincronización del programa con el *Start bit* modulado de una trama, esta sincronización consiste en preparar al programa para empezar la lectura de todos los bits modulados, esto se logra inicializando los registros encargados de este proceso. La subrutina Control utiliza el registro REGCON0 como bandera de control para indicar que la sincronización ya se efectúo, la sincronización solo se hace cuando se detecta el *Start bit* modulado y no se realiza mientras se lee la trama modulada, después de que esta ha sido leída y demodulada el programa queda a la espera de un nuevo *Start bit* y de una nueva sincronización.

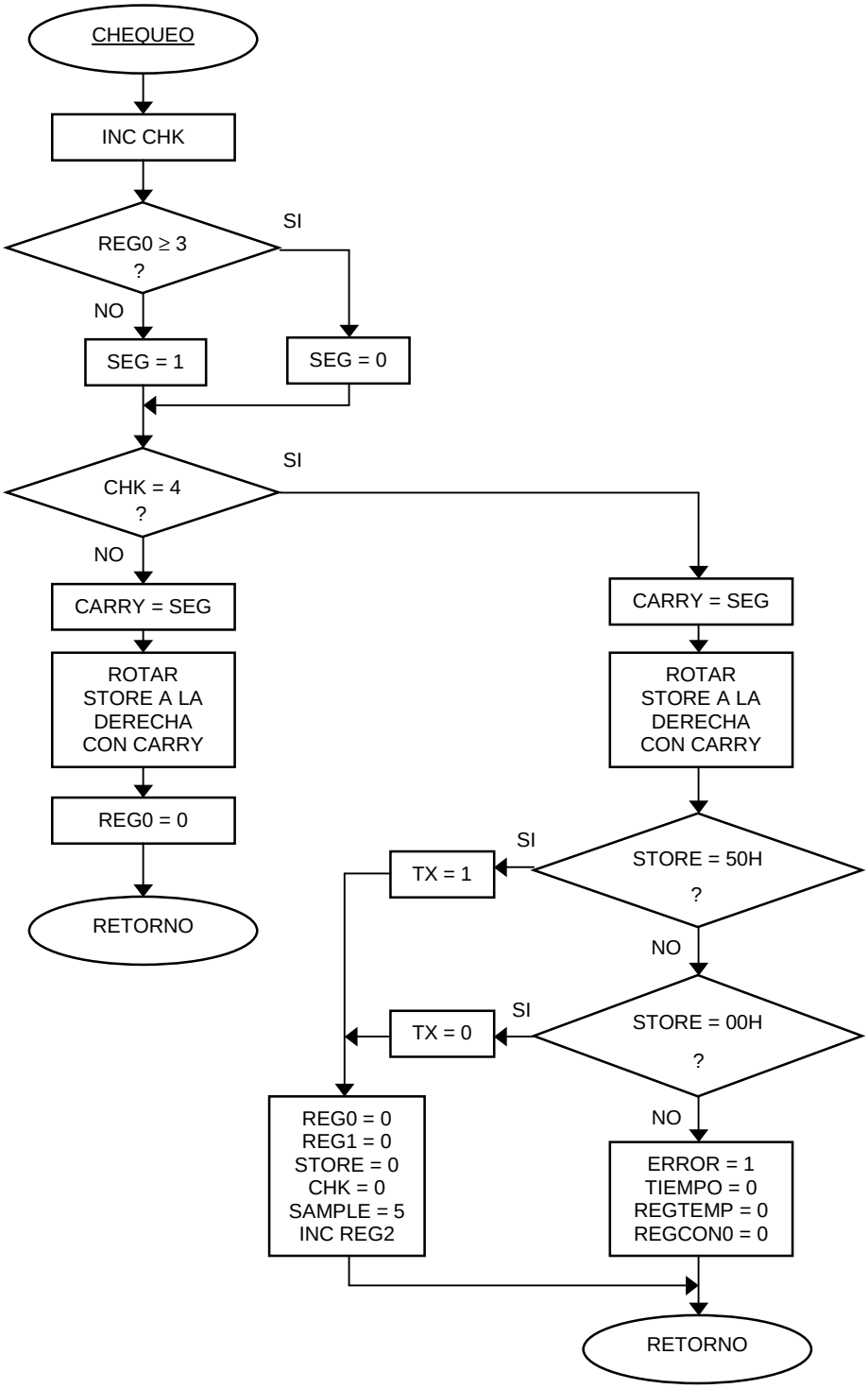
8.10.4.6 Subrutina Stop. Esta subrutina consiste en verificar que el ultimo bit leído (décimo bit) corresponda a un *Stop bit* valido, es decir, un uno lógico. La verificación se realiza revisando el contenido del bit seis (TX) del registro BUFFER, el cual contiene el estado lógico del ultimo bit demodulado. Si el estado de TX es uno indicara que se presento un *Stop bit* valido, pero si TX es cero se produce un error de comunicación, para lo cual se setea la bandera de ERROR y se clarea la bandera de TIEMPO obligando al programa a entrar de nuevo a la subrutina Tiempo de espera. En el caso de que el *Stop bit* sea valido se clarea la bandera de ERROR y se realiza el reinicio del programa, es decir, se prepara al programa para recibir un nuevo *Start bit* modulado.

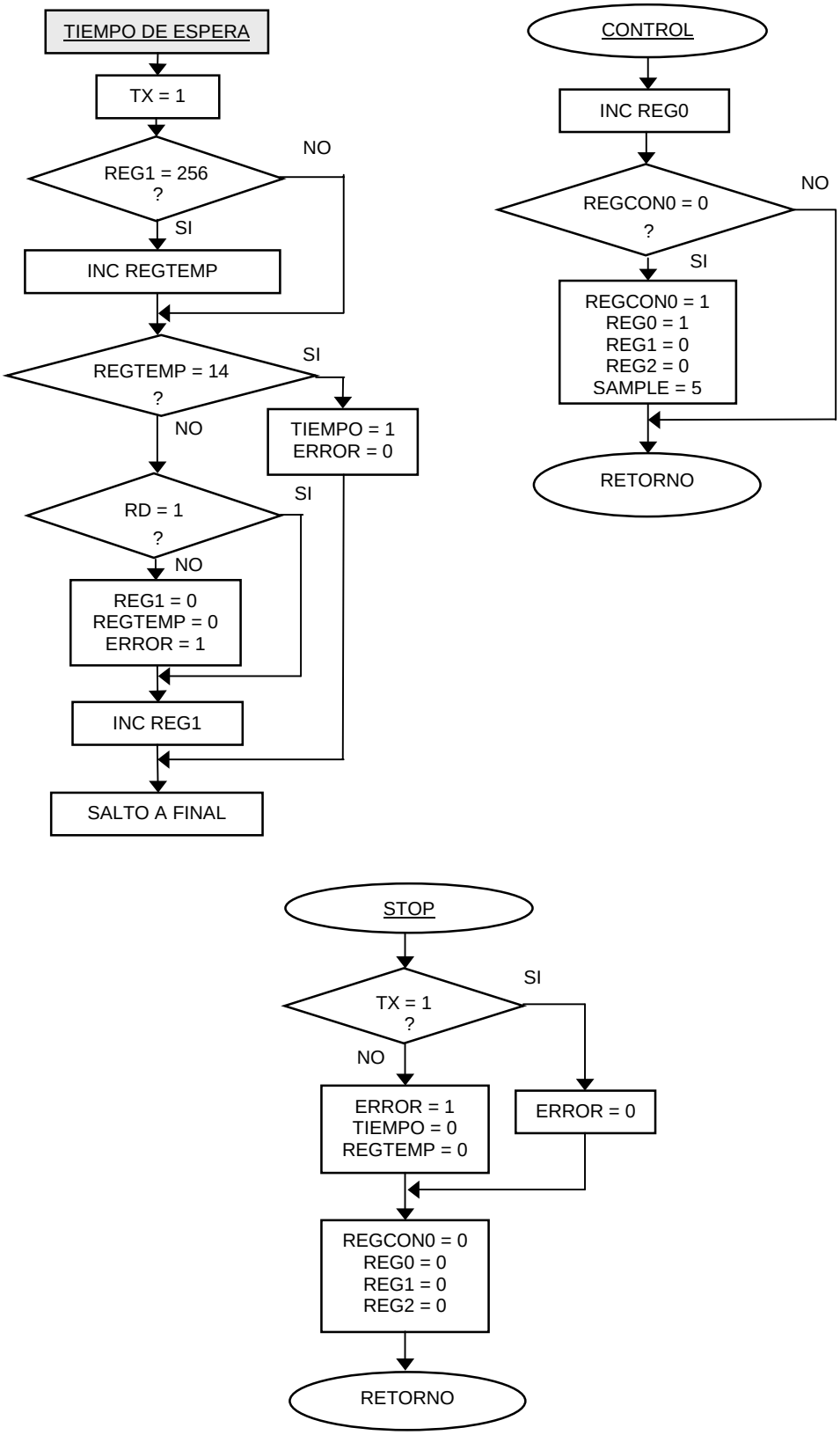
8.10.5 Diagrama de flujo del receptor. A continuación se presenta el diagrama de flujo de la rutina de servicio a la interrupción y de las subrutinas que fueron necesarias implementar en el desarrollo de la etapa de recepción para el módulo ASK.

RUTINA DE SERVICIO A LA INTERRUPCIÓN DEL RECEPTOR ASK



SUBRUTINAS


CHEQUEO
INC CHK
REG0 ≥ 3 ?
SI
NO
SEG = 1
SEG = 0
CHK = 4 ?
SI
NO
CARRY = SEG
ROTAR STORE A LA DERECHA CON CARRY
REG0 = 0
RETORNO
CARRY = SEG
ROTAR STORE A LA DERECHA CON CARRY
STORE = 50H ?
SI
TX = 1
NO
STORE = 00H ?
SI
TX = 0
REG0 = 0
REG1 = 0
STORE = 0
CHK = 0
SAMPLE = 5
INC REG2
NO
ERROR = 1
TIEMPO = 0
REGTEMP = 0
REGCON0 = 0
RETORNO


TIEMPO DE ESPERA
TX = 1
REG1 = 256 ?
NO
SI
INC REGTEMP
REGTEMP = 14 ?
SI
NO
TIEMPO = 1
ERROR = 0
RD = 1 ?
SI
NO
REG1 = 0
REGTEMP = 0
ERROR = 1
INC REG1
SALTO A FINAL
CONTROL
INC REG0
REGCON0 = 0 ?
NO
SI
REGCON0 = 1
REG0 = 1
REG1 = 0
REG2 = 0
SAMPLE = 5
RETORNO
STOP
TX = 1 ?
SI
NO
ERROR = 1
TIEMPO = 0
REGTEMP = 0
ERROR = 0
REGCON0 = 0
REG0 = 0
REG1 = 0
REG2 = 0
RETORNO

8.10.6 Programa en código fuente del receptor. A continuación se da el programa detallado, que se utilizo en el microcontrolador, con sus respectivos comentarios.

```
;-------------------------------------------------------------------------------------------------------------------
;                                        Demodulador ASK (Receptor)
;-------------------------------------------------------------------------------------------------------------------
;       ASKRX.ASM

        list p=16c622           ;Procesador
        include <p16c622.inc>

;       Al programar el micro tener en cuenta los fusibles de configuración
;       OSC = HS, WDT = OFF, PWRTE = OFF, CP = OFF, BODEN = OFF


;------------------------------------------------- Zona de Registros ---------------------------------------------


tmr_opt     equ     01h     ;Timer0 / option
status      equ     03h     ;
intcon      equ     0bh     ;
puertob     equ     06h     ;
buffer      equ     2ah     ;Registro que almacena cambios del puertob
reg0        equ     2bh     ;Contador de ceros
reg1        equ     2ch     ;Contador de interrupciones
reg2        equ     2dh     ;Contador de bits
regcon0     equ     2eh     ;Bandera de control de start bit
tempwr      equ     2fh     ;Temporal del reloj
store       equ     30h     ;Registro que almacena segmentos leídos
tiempo      equ     31h     ;Bandera de control del tiempo de espera
regtemp     equ     32h     ;Contador para determinar el tiempo de espera
chk         equ     33h     ;Contador de segmentos de bit leídos
sample      equ     34h     ;Contador de paquetes de dos muestras
seg         equ     35h     ;Registro que guarda el segmento actual
wtemp       equ     70h     ;Temporal de w
stemp       equ     71h     ;Temporal de status


;------------------------------------------------ Asignaciones de bit --------------------------------------------


rd          equ     7       ;Entrada serial (ASK)
tx          equ     6       ;Salida serial (RS-232)
err         equ     5       ;Aviso de error
gie         equ     7       ;Habilita int. general
toie        equ     5       ;Habilita int. por tmr0
toif        equ     2       ;Flag de int. por tmr0
rp0         equ     5       ;Página de memoria
sincro      equ     2       ;Sincronización del osciloscopio
```

```
;------------------------------------------------ Configuración del PIC ----------------------------------------------

        org     0
        goto    inicio          ;Inicio de programa principal

        org     4
        goto    inter           ;Vector de interrupción

        org     5               ;Inicio programa "inicio"

inicio  bsf     status,rp0      ;Va al banco1
        movlw   08h             ;Option, clock interno
        movwf   tmr_opt         ;
        movlw   80h             ;Configuración de puertos
        movwf   puertob         ;Puertob 7:entrada, 6-0:salida
        bcf     status,rp0      ;Retorno al banco0

        clrf    tmr_opt         ;Inicialización de registros
        clrf    puertob         ;
        clrf    reg0            ;
        clrf    reg1            ;
        clrf    reg2            ;
        clrf    regcon0         ;
        clrf    buffer          ;
        clrf    tempwr          ;
        clrf    store           ;
        clrf    tiempo          ;
        clrf    regtemp         ;
        clrf    chk             ;
        clrf    sample          ;
        clrf    seg             ;

        bsf     intcon,gie      ;Habilitación general de interrupciones
        bsf     intcon,toie     ;Habilita interrupción del timer0
        bsf     buffer,tx       ;Inicialización del puerto b
        bcf     buffer,sincro   ;

;------------------------------------------------------------ Loop ------------------------------------------------------

loop    goto    loop            ;Programa principal, loop infinito


;---------------------- RUTINA DE SERVICIO A LA INTERRUPCION DEL TMRO ----------------------


inter   movwf   wtemp           ;Push
        swapf   status,w        ;
        movwf   stemp           ;

        movf    tmr_opt,w       ;Ajuste de tiempo del timer0
        movwf   tempwr          ;
        btfss   tempwr,0        ;
        goto    time3e          ;


time3e  movlw   3eh             ;3eh -> 41,6 useg (aproximadamente)
        movwf   tmr_opt         ;Carga del timer0

princip movf    buffer,w        ;
        movwf   puertob         ;Actualización del puerto b
```

```
        btfss   tiempo,0        ;Se verifica si ya se ejecuto el tiempo de espera
        goto    time            ;

        btfss   regcon0,0       ;Mientras no se de un start bit, tx es 1 (reposo)
        goto    jump            ;

;---------------------------------------------------- Conversión ----------------------------------------------------


        movf    sample,w        ;
        subwf   reg1,w          ;Reg1 = sample?
        btfsc   status,2        ;Si reg1 = sample -> call check
        call    check           ;Chequeo del segmento leído
        goto    leer            ;

jump    bsf     buffer,tx       ;Estado de reposo de la línea


;---------------------------------------------------- Muestreo ----------------------------------------------------


leer    btfss   puertob,rd      ;Lectura de la entrada serial
        call    control         ;
        incf    reg1,f          ;Incrementa contador de interrupciones

stopbit movlw   0ah             ;Verificación del stop (bit 10)
        subwf   reg2,w          ;
        btfsc   status,2        ;Reg2 = 10?
        call    stop            ;Chequea si el bit 10 fue un 1
        goto    final           ;


;---------------------------------------------- Subrutina tiempo de espera ----------------------------------------


time    bsf     buffer,tx       ;Mientras se ejecuta el tiempo de espera, tx es 1 (reposo)
        movlw   0ffh            ;Tiempo de espera de aproximadamente 150mseg
        subwf   reg1,w          ;41.6useg * 256 = 10.6mseg
        btfsc   status,2        ;
        incf    regtemp,f       ;Se cuentan ciclos de 10.6mseg

        movlw   0eh             ;
        subwf   regtemp,w       ;10.6mseg * 14 = 149.1mseg
        btfss   status,2        ;Regtemp = 14 -> tiempo = 1 y error = 0
        goto    again           ;
        bsf     tiempo,0        ;Cumplidos los 150mseg se setea la bandera de tiempo
        bcf     buffer,err      ;Se clarea la bandera de error
        goto    final           ;

again   btfsc   puertob,rd      ;Lectura de la entrada serial
        goto    unor            ;
        clrf    reg1            ;Si rd es cero durante el tiempo de espera
        clrf    regtemp         ;se inicia de nuevo el conteo
        bsf     buffer,err      ;
unor    incf    reg1,f          ;Incremento del contador de interrupciones
```

```
;---------------------------------------------------------- Final ----------------------------------------------------------

final   bcf     intcon,toif     ;Se clarea la bandera del tmr0
        swapf   stemp,w         ;Pop
        movwf   status          ;
        swapf   wtemp,f         ;
        swapf   wtemp,w         ;
        retfie                  ;

;------------------------------------------------------ Subrutina check ------------------------------------------------------


check   movlw   05h             ;Sample = sample + 5
        addwf   sample,f        ;
        incf    chk,f           ;

        movlw   03h             ;Segmento
        subwf   reg0,w          ;
        bcf     seg,0           ;Se asume seg = 0
        btfss   status,0        ;Si reg0 >= 3 -> seg = 0
        bsf     seg,0           ;

        movlw   04h             ;
        subwf   chk,w           ;Chk = 4?
        btfsc   status,2        ;Si chk = 4 -> Convierte a RS-232
        goto    equal4          ;

rotar   bcf     status,0        ;
        btfsc   seg,0           ;Carry = seg
        bsf     status,0        ;
        rrf     store,f         ;Rotación de store a la der. con carry
        clrf    reg0            ;Inicio del próximo segmento
        return                  ;Retorno

equal4  bcf     status,0        ;
        btfsc   seg,0           ;Carry = seg
        bsf     status,0        ;
        rrf     store,f         ;Ultima rotación de store a la der.

uno     movlw   50h             ;
        subwf   store,w         ;Store = 50h?
        btfss   status,2        ;Si store = 50h -> tx = 1
        goto    cero            ;
        bsf     buffer,tx       ;Conversión a RS-232 para un uno
        goto    next            ;

cero    movf    store,w         ;Store = 00h?
        btfss   status,2        ;Si store = 00h -> tx = 0
        goto    fail            ;
        bcf     buffer,tx       ;Conversión a RS-232 para un cero

next    incf    reg2,f          ;
        clrf    reg0            ;Iniciación de bit
        clrf    reg1            ;
        clrf    store           ;
        clrf    chk             ;
        movlw   05h             ;
        movwf   sample          ;
        return                  ;Retorno
```

```
fail    bsf     buffer,err      ;Aviso de error si el bit leído no corresponde
        clrf    tiempo          ;a un uno o un cero en ASK
        clrf    regtemp         ;
        clrf    regcon0         ;
        return                  ;Retorno

;------------------------------------------------- Subrutina stop ---------------------------------------------------

stop    bcf     buffer,err      ;Se asume error =  0
        btfsc   buffer,tx       ;Chequeo del bit 10 para definir si hay error
        goto    valido          ;tx = 1 para el bit 10, stop valido
        bsf     buffer,err      ;tx = 0 para el bit 10, stop no valido
        clrf    tiempo          ;Se inicia de nuevo el tiempo de espera
        clrf    regtemp         ;
valido  clrf    regcon0         ;
        clrf    reg0            ;
        clrf    reg1            ;
        clrf    reg2            ;
        movlw   04h             ;Final de la trama, se permite que se vuelva a
        xorwf   buffer,f        ;sincronizar con el osciloscopio (bit 2 del puertob)
        return                  ;

;------------------------------------------------- Subrutina control ------------------------------------------------

control incf    reg0,f          ;
        movf    regcon0,w       ;Chequea si regcon0 = 0
        btfsc   status,2        ;
        call    sinc            ;Regcon0 = 0 -> Sincronización con Start bit
        return                  ;

;------------------------------------------------- Sincronización ---------------------------------------------------

sinc    movlw   04h             ;Señal de sincronización del osciloscopio
        xorwf   buffer,f        ;con el inicio de la trama (bit 2 del puertob)
        clrf    reg0            ;Sincronización con el start bit
        clrf    reg1            ;
        clrf    reg2            ;
        movlw   05h             ;
        movwf   sample          ;
        incf    reg0,f          ;
        bsf     regcon0,0       ;Se deshabilitan futuras sincronizaciones
        return                  ;por un 0. Se restablece con el Stop bit

;------------------------------------------------------------------------------------------------------------------------
        end
```

8.11 HARDWARE DE LOS MODULOS MANCHESTER, FSK , PSK Y ASK

Los cuatro módulos que realizan modulación de señales digitales emplean la misma distribución de hardware (circuito eléctrico), ya que el proceso eléctrico de la señal digital es siempre el mismo.

La única diferencia entre los módulos es el software usado en el microcontrolador, que se encarga de llevar a cabo la modulación o demodulación correspondiente.

8.11.1 Funcionamiento del Transmisor. Como se aprecia en el diagrama esquemático de la Figura 32, la entrada digital serial maneja niveles de voltaje del protocolo RS-232, presentes en el pin 3 (TD) del conector DB9 de entrada, estos deben ser llevados a niveles de voltaje TTL para que puedan ser interpretados por el microcontrolador, para realizar esta operación la señal pasa por un *line driver / receiver* (driver y receptor de línea) que en este caso es un MAX-232, este se encarga de cambiar los niveles lógicos, así un voltaje positivo (+10V) que representa un cero lógico RS-232 se convierte a 0V TTL y un voltaje negativo (-10V) que representa un 1 lógico se convierte a +5V TTL.

Teniendo los niveles de voltaje correctos la señal es enviada al pin 13 (RB7) del microcontrolador PIC16C622, este ultimo se encarga de procesarla (según el programa que resida en su memoria), de acuerdo al sistema de modulación

de cada módulo y generar así la señal de salida correspondiente, la cual se obtiene del pin 12 (RB6) del PIC16C622.

La señal de salida del microcontrolador debe convertirse a una señal del tipo *RZ Polar*, la cual debe estar presente en la línea entre los módulos de transmisión y recepción, para lograr esto se reutiliza el MAX-232 de manera que convierta los niveles TTL a niveles de doble polaridad, pero se debe tener en cuenta que estos niveles son del protocolo RS-232 que maneja lógica negativa, es decir, un voltaje positivo representa un 0 binario y un voltaje negativo representa un 1 binario, por esta razón la señal de salida debe invertirse, para lo cual se utiliza una compuerta inversora 7414, y de esta manera después de pasar por el MAX-232 la señal usara lógica positiva, así un voltaje positivo representara un 1 binario y un voltaje negativo representara un 0 binario, esta señal será la que se envié a la línea de transmisión por medio del pin 2 (RD) del conector DB9 de Salida .

El transmisor tiene dos diodos LED, estos indican el estado del módulo de la siguiente manera: el LED *D2* (ERROR) se encarga de indicar cuando se presenta un error en la comunicación entre el Computador y el módulo, y el LED *D3* (POWER) es el indicador de encendido.

Existen además ocho puntos de prueba, que son los que permiten observar con ayuda del osciloscopio las distintas señales asociadas al tipo de modulación que usa el módulo, de la siguiente forma:

- IN_PC: En este punto de prueba se observa la trama de datos que proviene del Computador.

- LINEA: Este permite observar la señal *RZ polar* presente en la línea entre los módulos de Transmisión y Recepción (Señal Modulada).

- GND: Tierra (se envía al terminal común del Osciloscopio).

- SINCRO: Esta señal es un pulso de sincronización generado para poder sincronizar la trama con el Osciloscopio, permitiendo así una observación más estable (se conecta a la entrada de sincronización externa del Osciloscopio). El Osciloscopio debe configurarse para usar sincronización por flanco de subida.

- OSC1 y OSC2: En estos puntos se puede ver la señal portadora y su desplazamiento según el tipo de modulación, es decir, las dos posibles formas de representar los estados lógicos.

- CODIF: Es la señal de salida del microcontrolador, es decir, la señal modulada pero en niveles TTL.

- LECTURA: Es la señal que proviene del Computador convertida a niveles TTL, esta es la que se emplea como entrada del microcontrolador.

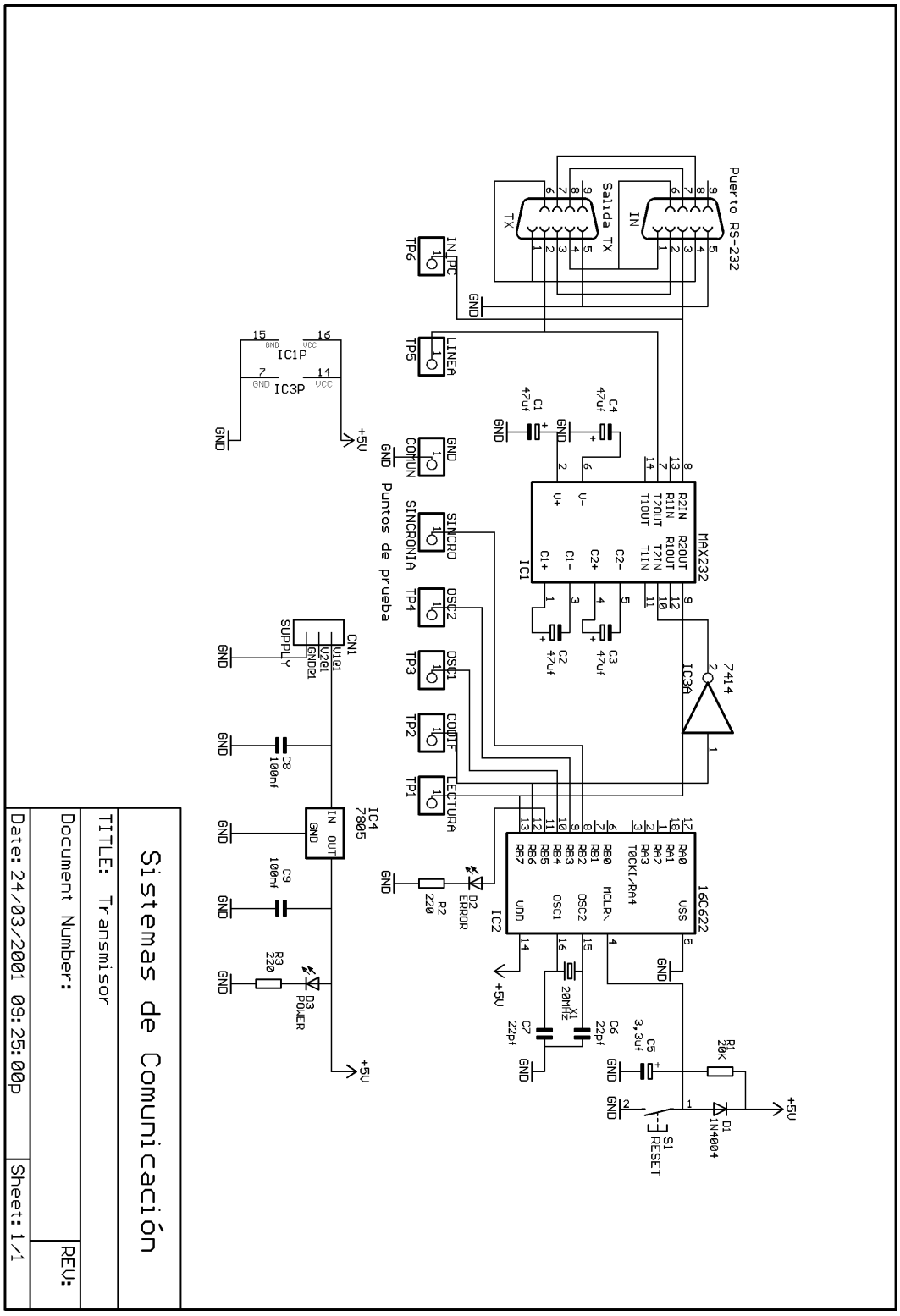
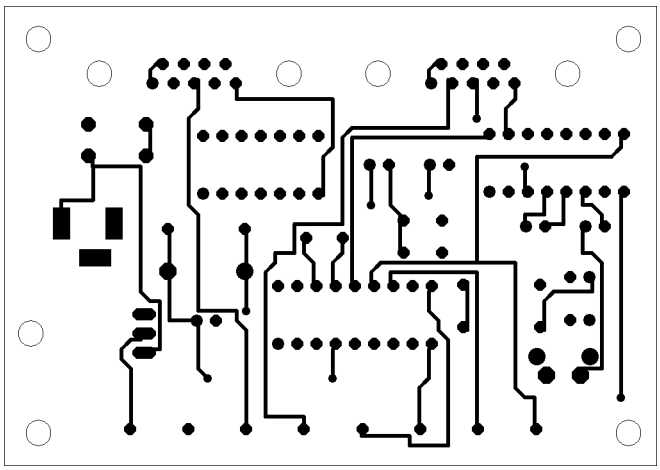
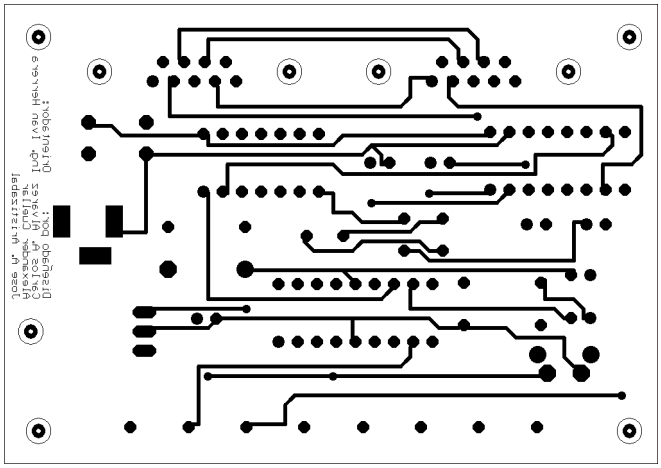


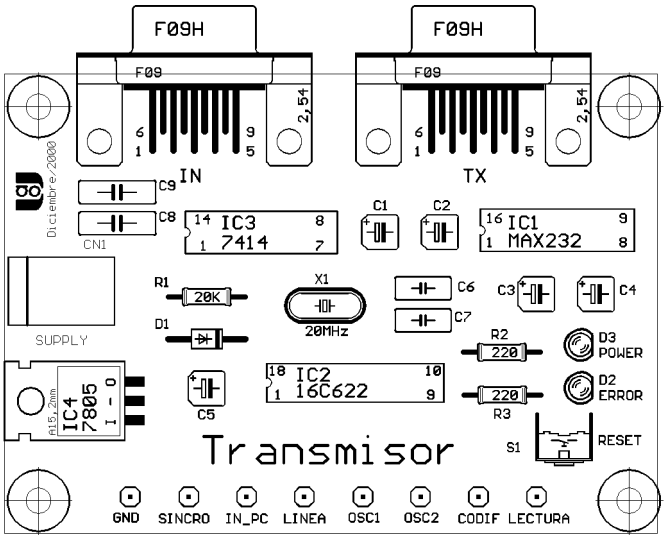
Figura 32 *Diagrama esquemático del transmisor Manchester, FSK, PSK y ASK*



(a) Vista Superior



(b) Vista Inferior



(c) Vista de componentes

Figura 33 *Circuito Impreso del transmisor Manchester, FSK, PSK y ASK*

8.11.2 Funcionamiento del Receptor. De acuerdo al diagrama esquemático de la Figura 34, la línea que contiene la señal modulada del tipo *RZ polar* es el pin 2 (RD) del conector DB9 de entrada, esta señal debe convertirse a niveles TTL para su procesamiento, por lo que se utiliza un *line driver / receiver* como el MAX-232, pero se debe tener en cuenta que este además de convertir los niveles de voltaje también trabaja como una compuerta inversora, de esta manera la señal de entrada que utiliza lógica positiva quedaría convertida a una señal TTL con lógica negativa, así que a la salida del MAX-232 se debe invertir de nuevo la señal, por lo que se usa un 7414 (inversor) para devolverla a su estado lógico inicial.

La señal es entonces enviada al pin 13 (RB7) del microcontrolador PIC16C622 el cual se encarga de realizar el proceso de demodulación (según el programa que resida en su memoria) para recuperar la trama de datos original, estos datos conforman la salida, la cual se obtiene del pin 12 (RB6) del PIC16C622. Los datos se convierten de nuevo al protocolo RS-232 con la ayuda del MAX-232, para por ultimo ser enviados a un Computador que permita visualizarlos.

El receptor incorpora dos diodos LED, estos indican el estado del módulo de la siguiente forma: el LED *D2* (ERROR) se encarga de indicar cuando se presenta un error en la comunicación entre el módulo de transmisión y el de recepción, y el LED *D3* (POWER) es el indicador de encendido.

En el módulo de recepción existen ocho puntos de prueba, los cuales permiten ver con ayuda del Osciloscopio las señales asociadas al proceso de demodulación, los puntos son los siguientes:

- OUT_PC: En este punto se observa la trama de datos original, que fue enviada por el módulo transmisor y recuperada por el receptor.

- LINEA: Este permite observar la señal *polar RZ* presente en la línea entre los módulos de Transmisión y Recepción (Señal Modulada).

- GND: Tierra (se envía al terminal común del Osciloscopio).

- SINCRO: Esta señal es un pulso de sincronización generado para poder sincronizar la trama con el Osciloscopio, permitiendo así una observación más estable (se conecta a la entrada de sincronización externa del Osciloscopio). El Osciloscopio debe configurarse para usar sincronización por flanco de subida.

- OSC1 y OSC2: No se usan.

- DECODIF: Es la señal de salida del microcontrolador, es decir, la señal demodulada pero en niveles TTL.

- LECTURA: Es la señal que proviene de la línea entre transmisor y receptor convertida a niveles TTL, esta es la que se emplea como entrada del microcontrolador.

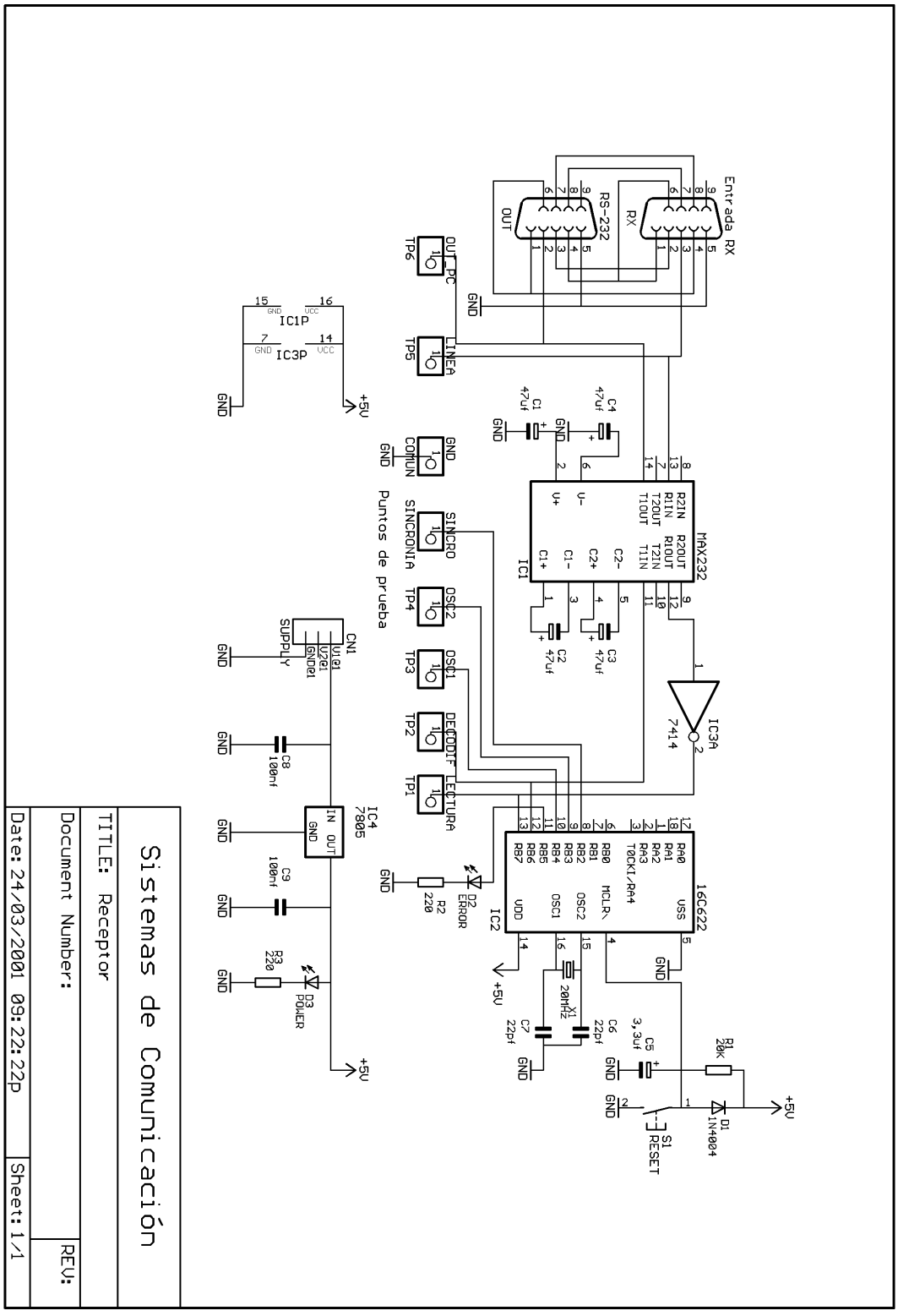
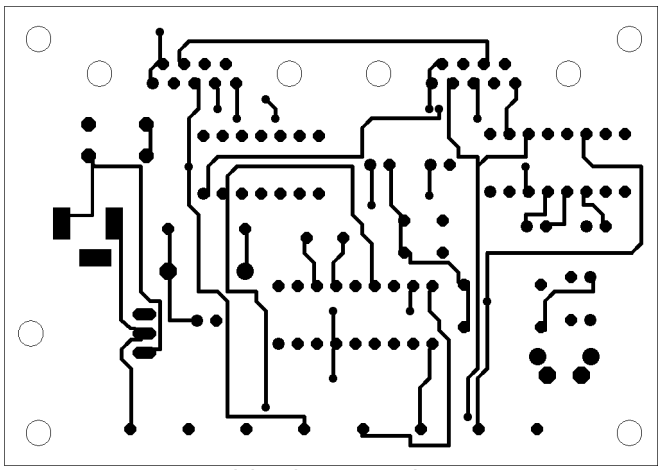
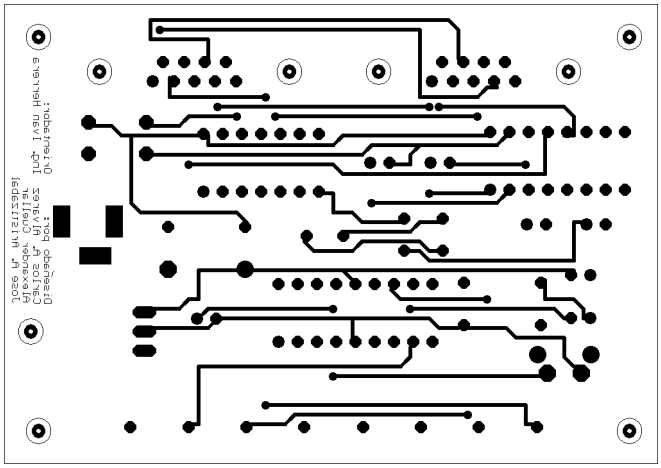


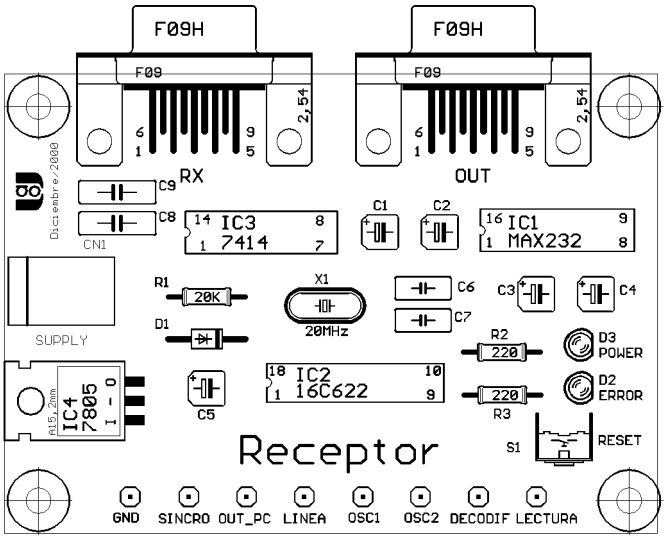
Figura 34 *Diagrama esquemático del receptor Manchester, FSK, PSK y ASK*



(a) Vista Superior



(b) Vista Inferior



(c) Vista de componentes

Figura 35 *Circuito Impreso del receptor Manchester, FSK, PSK y ASK*

8.12 PRINCIPIOS DE DISEÑO PARA LOS MODULOS PCM Y DELTA

De la misma manera que en los módulos de codificación de señales digitales, la transmisión en cada uno de los módulos utiliza como programa principal un bucle infinito, el cual es interrumpido por el Timer para obtener una muestra de la línea de entrada, que se procesa para actualizar el estado de la línea de transmisión entre transmisor y receptor, (Ver Figura 36).

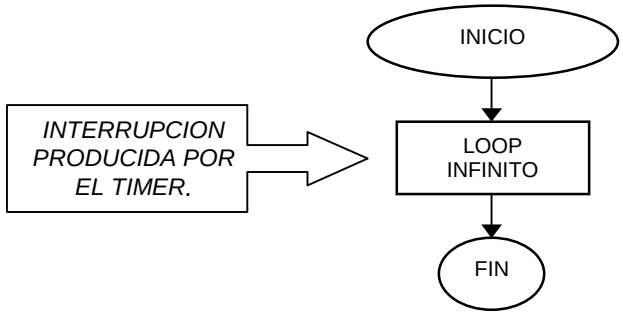


Figura 36 *Programa principal de las etapas de transmisión PCM y DELTA*

En el caso del módulo PCM el bucle infinito es interrumpido por la acción de dos Timers ya que se requiere producir un tipo de onda especifico (PAM).

En la recepción simplemente se espera a que el registro de recepción de la USART se llene, es decir se recibe un dato, para luego ser procesado.

Para estos dos módulos se utiliza un generador de funciones (XR2206) que proporciona las señales de entrada, este genera frecuencias entre 15Hz y 30Hz para el Módulo DELTA, y frecuencias entre 50Hz y 100Hz para el Módulo PCM.

8.13 MODULO PCM

Con este módulo se transmitirá una señal análoga a través de un sistema binario PCM.

El primer paso en la conversión de una señal análoga a una señal PCM (digital) es el proceso de conversión de análogo a PAM (Modulación por Amplitud de Pulsos).

La señal PAM consiste en la conversión de una señal análoga a una señal tipo pulso donde la amplitud del pulso contiene la información análoga.

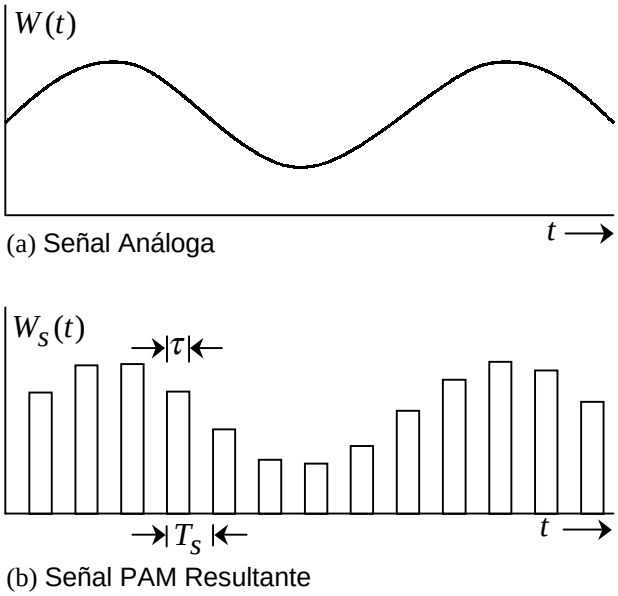


Figura 37 *Formas de onda de la señal Análoga y la señal PAM Resultante*

Como se aprecia en la Figura 37, se captura una muestra de la señal análoga cada T_S periodos de tiempo produciendo un pulso cuya amplitud será aproximadamente igual al de la señal original en ese instante. El ancho del pulso τ es también llamado *apertura*, entonces τ / T_S determina la ganancia de

la señal recuperada, la cual es pequeña si τ es menor con relación a T_s. A la relación τ / T_s se le llama ciclo de trabajo *d*, donde $\tau \leq T_s$.

Para generar la señal PAM del módulo PCM se utilizan dos interrupciones, una producida por el Timer 0 y la otra por el Timer 2, que son temporizadores de ocho bits. De esta manera el Timer 0 es el encargado de capturar la muestra de la señal análoga con ayuda del conversor análogo-digital cada T_s periodos de tiempo, teniendo en cuenta que el generador de funciones posee una frecuencia máxima de 100Hz, se determina que la mínima frecuencia de muestreo es de:

$$f_S \geq 2 \times f\,(Hz) \Rightarrow f_{S\min} = 2 \times 100 = 200Hz$$

se decide entonces utilizar una frecuencia de muestreo de 1KHz, es decir 10 veces la frecuencia más alta de la señal análoga, esto permite reconstruir una señal casi idéntica a la original. Esta frecuencia de muestreo define la interrupción del Timer 0, de la siguiente manera:

$$T_S = \frac{1}{f_S} = \frac{1}{1000Hz} = 1ms$$

lo que indica que se tomara una muestra cada 1mseg, por lo que se configura el Timer 0 para que produzca una interrupción cada 1mseg.

La interrupción del Timer 2 es utilizada para ilustrar el ciclo de trabajo de la señal PAM, en el modulo PCM este ciclo es igual a uno (1), ya que la *apertura* τ es igual a T_s, es decir se aprovecha el 100% del pulso, gráficamente se apreciaría ya que no existiría separación entre los pulsos PAM. Pero a fin de

mostrar una señal más acorde con los textos sobre comunicaciones se decide generar una señal PAM con un ciclo de trabajo de 1/3, y con una *apertura* de:

$$d = \frac{\tau}{T_S} = \frac{1}{3} \Rightarrow \tau = \frac{T_S}{3} = \frac{1ms}{3} = 333{,}3\mu s$$

esto permite ver gráficamente una sucesión de pulsos separados aproximadamente cada 666,6µs, si se quisiera recuperar la señal original de esta señal PAM se observaría que la amplitud de la señal recuperada es mucho menor que el de la señal inicial, esto debido a que solo se estaría aprovechando el 33,3% del pulso que contiene la información de la muestra; pero como ya se menciono antes esto se hace solo con el fin de ilustrar el ciclo de trabajo en una señal PAM. Es entonces cuando el Timer 2 se configura de tal manera que produzca una interrupción aproximadamente a 333,3µs después de haberse producido la interrupción del Timer 0, esta interrupción se encargara de poner en ceros el puerto del microcontrolador en el cual se encuentra la palabra digital correspondiente a la muestra tomada durante la interrupción del Timer 0 de esta manera el conversor digital-análogo que reciba los datos del puerto reproducirá la señal PAM.

Después de realizar la operación de muestreo, la cual se encarga de generar la señal PAM, se realiza una operación de cuantización, es decir, cada muestra de la señal análoga se convierte a un entero binario de *n* bits (palabra digital), cuyo valor será la mejor aproximación de la muestra. Estas dos primeras operaciones son realizadas por el conversor análogo-digital de diez bits de resolución, incorporado en el microcontrolador, teniendo en cuenta la función de transferencia de la Figura 38.

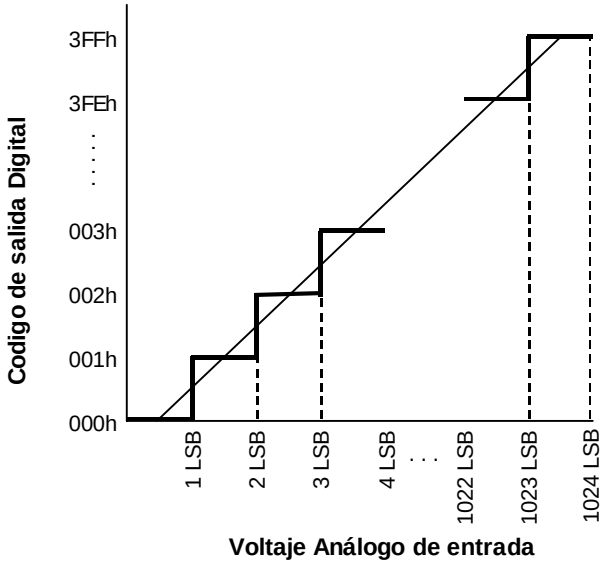


Figura 38 *Función de transferencia (Niveles de cuantización) del conversor análogo-digital de diez bits*

En la grafica se observa como el voltaje de entrada esta referido al peso del bit menos significativo (*1LSB*), el cual es definido por la resolución del conversor y el voltaje análogo de referencia, así $1LSB = Vref / 2^N$, donde *N* es el numero de bits de resolución; el módulo PCM utiliza para el ADC un voltaje de referencia de +5V.

Por ultimo para la generación de la señal PCM, la palabra digital necesita ser convertida a una forma serial para ser transmitida, de esto se encarga un conversor Paralelo a Serial como una USART, la cual también esta incorporada en el microcontrolador, para determinar la mínima tasa de transferencia de bits que esta debe manejar se debe tener en cuenta que se utilizara una trama PCM de ocho bits más un bit de señalización, por lo tanto:

$$R = n \times f_S = 9 \times 1000 = 9Kbaudios$$

Donde *n* es el numero de bits, f_s es la frecuencia de muestreo utilizada y *R* es la mínima tasa de transferencia. La USART del microcontrolador tiene una

mínima tasa de transferencia de 19.53Kbaudios que esta predeterminada por la velocidad de operación que es de 20MHz. Esto indica que al utilizar una velocidad mayor de transmisión se garantiza que no se alterara el ancho de banda del modulo PCM.

El ancho de banda *B* de la señal análoga que entra al sistema PCM esta determinado por la frecuencia de muestreo de la siguiente manera:

$$B = \frac{f_S}{2} = \frac{1000}{2} = 500Hz$$

esto quiere decir que la frecuencia máxima de la señal análoga de entrada debe ser de 500Hz, una frecuencia mayor causaría que la señal se distorsione (*Aliasing*).

El conversor análogo-digital del microcontrolador entrega una palabra digital de diez bits, y como ya se había mencionado se utiliza una trama PCM de ocho bits por lo que se hace necesario usar un factor de conversión para obtener una palabra digital con la resolución apropiada, así se usa:

$$D_8 = \frac{D_{10}}{4}$$

donde D_8 es la palabra digital de ocho bits, y D_{10} es la palabra digital de diez bits originada por el conversor análogo-digital.

Este factor de conversión modifica los niveles de cuantización del conversor análogo-digital, como se ve en la Figura 39.

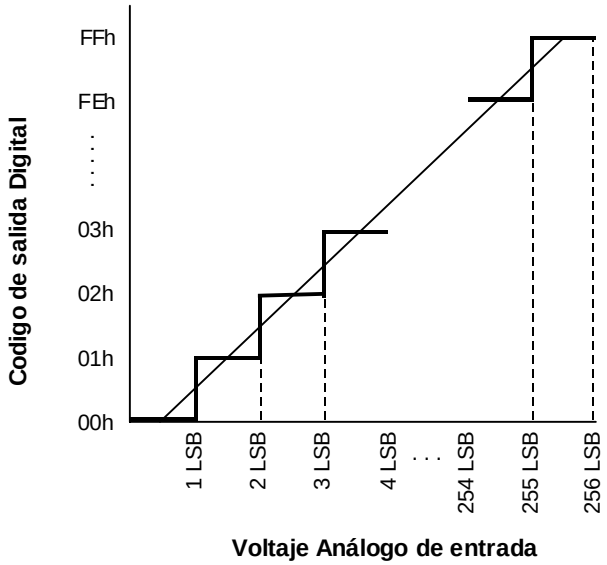


Figura 39 *Función de transferencia (Niveles de Cuantización) modificada a ocho bits del conversor análogo-digital*

La USART utiliza comunicación sincrónica, es decir, los datos son enviados junto con una señal de sincronismo, como se aprecia en la Figura 40.

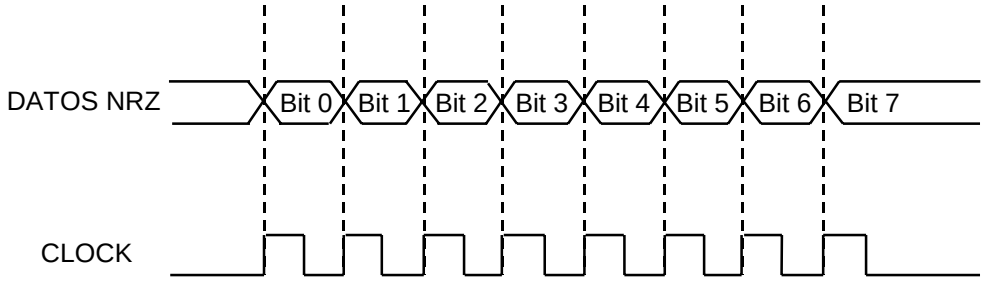


Figura 40 *Forma de onda Sincrónica PCM*

De manera que las líneas de transmisión entre los módulos de transmisión y recepción se reducen a solo tres (Data, Clock y Ground). La palabra digital transmitida corresponderá entonces a una muestra de la señal análoga.

Posterior al desarrollo del módulo PCM se incluyo una segunda señal de entrada la cual pasa por el mismo proceso que la primera, es decir, se tienen dos canales de entrada análogos. Pero, para transmitir cada una de las dos señales PCM, se utiliza multiplexación por división de tiempo (TDM), lo cual

consiste en intercalar cada una de las muestras de las entradas análogas para que la información pueda ser transmitida serialmente a través de un solo canal de comunicación.

Es necesaria la sincronización de la trama TDM para que los datos multiplexados puedan ser clasificados y dirigidos al canal de salida apropiado en el receptor.

La sincronización de la trama TDM se hace incluyendo un noveno bit de señalización en la trama PCM, este es el encargado de indicar el origen y destino de la misma, como se ve en la Figura 41.

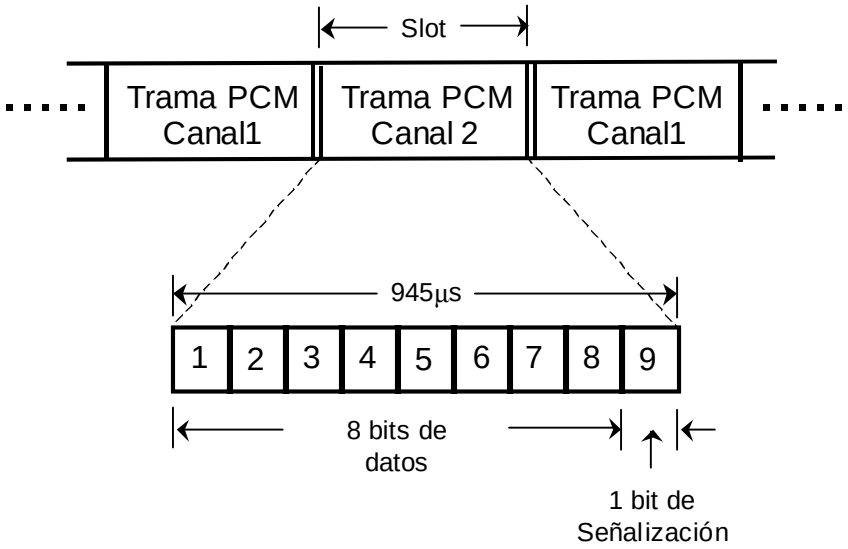


Figura 41 *Formato de una trama TDM-PCM*

En la recepción, la señal PCM es decodificada de nuevo a una señal análoga usando un conversor digital-análogo, los datos son entonces convertidos por este a una aproximación del valor de la muestra de la señal análoga.

En resumen el proceso de transmisión y recepción utilizado en el módulo PCM puede ser descrito como se ve en la Figura 42.

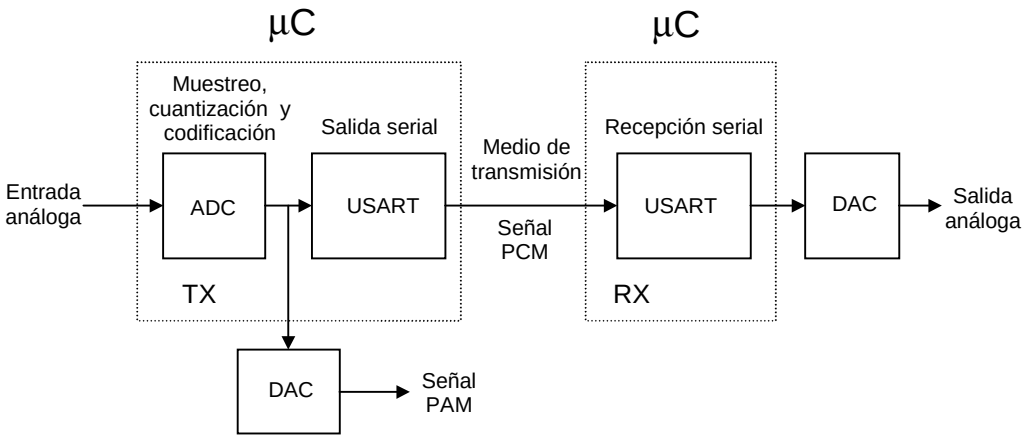


Figura 42 *Diagrama de bloques de los módulos de transmisión y recepción PCM*

En la gráfica, el transmisor, realiza las operaciones de muestreo, cuantización y codificación de la señal análoga utilizando un conversor análogo-digital, luego la muestra codificada es transmitida serialmente utilizando una USART. Un conversor digital-análogo se encarga de generar una señal PAM con la muestra codificada.

En la etapa de recepción, los datos llegan de nuevo a una USART que los convierte de serie a paralelo, para luego ser enviados a un conversor digital-análogo que reproduce una aproximación de la señal análoga.

8.13.1 Software del transmisor. El software del transmisor es quien realiza la modulación PCM de las entradas análogas.

El programa principal es un loop infinito, el cual solo es interrumpido por el desbordamiento del Timer 0 o por la igualación del Timer 2 con el registro PR2, ambos casos generan una interrupción. La interrupción del Timer 0 se produce cada 1ms y la interrupción del Timer 2 se produce 333µs después de presentarse la del Timer 0. Cualquiera de estas interrupciones dirige el programa a la dirección de memoria 0004H donde se encuentra el vector de interrupciones. En este sitio se ubica un salto que apunta a la dirección donde se encuentra la rutina de servicio a las interrupciones. Estas rutinas son las que finalmente se encargan del procesamiento de las entradas análogas y de la generación de las señales de salida.

Al entrar a la rutina de servicio a las interrupciones se verifica quien la genera, de esta manera se sabe que rutina se debe ejecutar para atender la interrupción. Si la interrupción es generada por el Timer 0 se ejecuta un salto que apunta al sitio donde se encuentra la rutina que la atenderá. La función de esta rutina es tomar una muestra de cada una de las dos entradas análogas utilizando el conversor análogo-digital y transmitirla con la ayuda de la USART. De acuerdo a la posición de un selector ubicado en un pin del puerto D, se envía una de las muestras tomadas al puerto B, de esta manera un conversor digital-análogo construye la señal PAM. Durante esta rutina se inicia el conteo de tiempo del Timer 2.

Cuando la interrupción es generada por el Timer 2, que debe ser 333µs después de producirse la interrupción del Timer 0, se salta al lugar donde se encuentra la rutina que la atenderá. La función de esta rutina es generar el ciclo

de trabajo de la señal PAM, esto se logra clareando el puerto B. Esta rutina desactiva el conteo del Timer 2.

A continuación se hace una descripción detallada de las rutinas implementadas para el desarrollo del modulador PCM.

8.13.1.1 Programa principal. En esta parte del programa se configura el Timer 0, el cual utiliza la fuente de reloj interna del microcontrolador y el prescaler en 1:32, se carga el registro PR2 para que el Timer 2 contabilice 333μs con un prescaler de 16, se habilita la interrupción del Timer 2 para que esta se pueda producir, se programa el puerto B como salida y el bit cero del puerto D como entrada, se configura el conversor análogo-digital con referencias de cinco voltios (+5V) y tierra, el resultado queda justificado a la derecha, se configura la USART a 19,53Kbaudios, en modo maestro y sincrónico, utilizando 9 bits de datos, se habilita la transmisión y se habilita el puerto serial, por ultimo se inicializan todos los registros que se utilizan en el programa y se habilita la interrupción del Timer 0. Esto es realizado únicamente al inicio del programa y no se vuelve a entrar en el mientras el programa este corriendo. Luego el programa ingresa a un loop infinito, es decir, se ejecuta un salto que apunta a la dirección de memoria donde se encuentra el mismo salto, este lazo cerrado que se produce solo se interrumpe por la acción de interrupción del Timer 0 y del Timer 2.

8.13.1.2 Rutina de servicio a la interrupción del Timer 0 y del Timer 2. Esta rutina inicialmente ejecuta un PUSH, el cual se encarga de almacenar el registro de trabajo W y las banderas de estado de la ALU como Carry, Digit Carry y Cero que se encuentran en el registro STATUS.

Después se verifica la fuente de la interrupción, es decir, si la interrupción fue producida por el Timer 0 o por el Timer 2, de esta manera se sabe cual es la rutina que debe atender la interrupción. Para realizar la verificación se comprueba el estado de la bandera TOIF, esta se pone en uno cuando la interrupción es generada por el Timer 0, pero si es cero esto indicara que la interrupción es producida por el Timer 2.

Si la interrupción es generada por el Timer 0, se ejecuta un salto que apunta al sitio donde esta la rutina que la atenderá, lo primero que se realiza al entrar a la rutina es el ajuste de tiempo del Timer 0, este ajuste se realiza para cargar el Timer 0 correctamente y de esta manera calibrar el tiempo en que se presentara la próxima interrupción. Después se activa el Timer 2 para que inicie el conteo de 333μs. Luego se procede a tomar las muestras de las dos entradas análogas, esto se realiza con la ayuda del conversor análogo-digital de ocho canales incorporado en el microcontrolador, de esta manera se toma primero la muestra del canal cero (SEÑAL 1) y luego la del canal uno (SEÑAL 2), el procedimiento para tomar las muestras se inicia con la elección del canal de entrada, después se activa el conversor análogo-digital, esto inicia la adquisición de la muestra para lo cual existe un tiempo requerido, este tiempo es contabilizado por la subrutina Retardo, esta subrutina ejecuta un retardo de

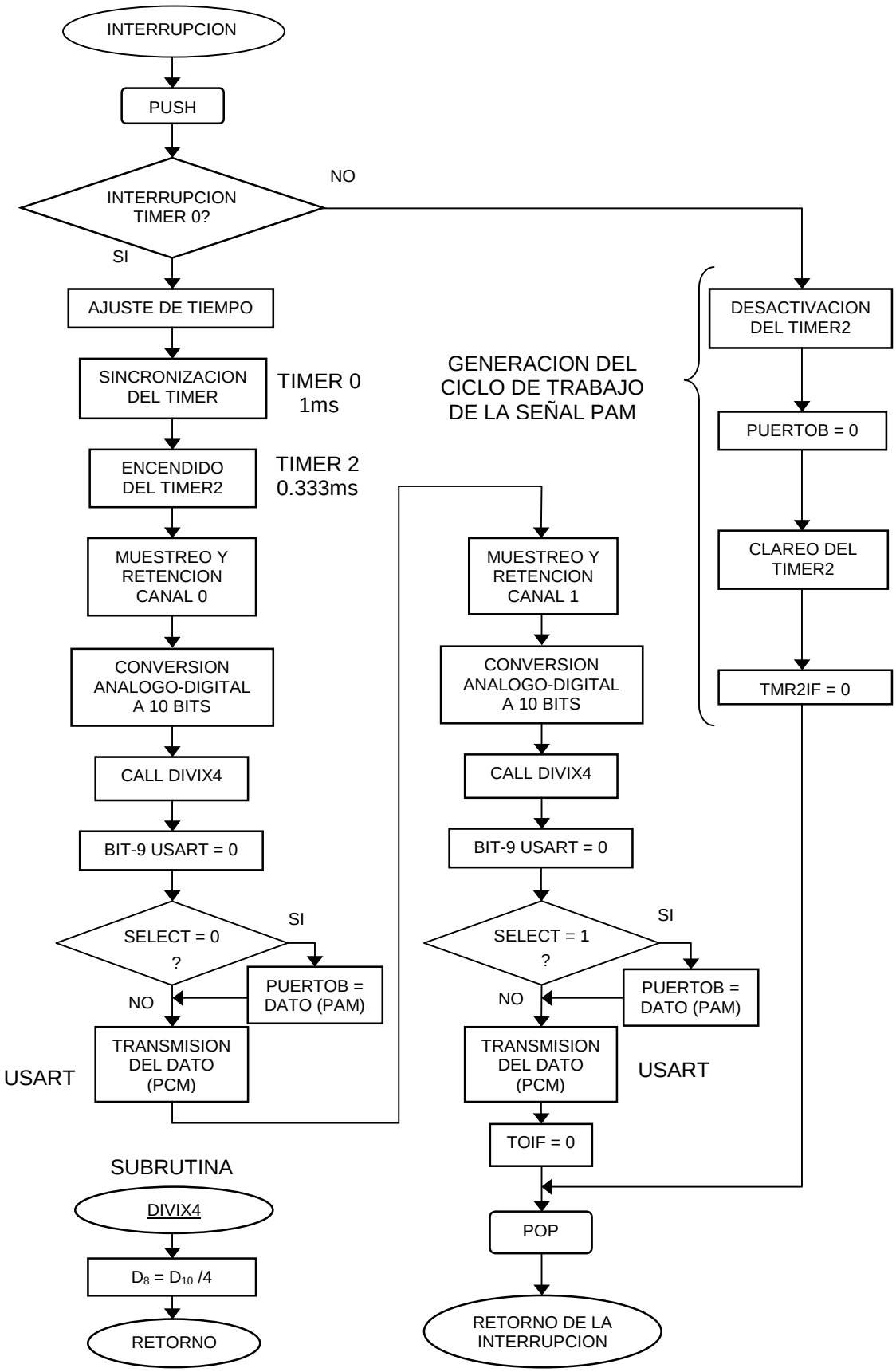
aproximadamente 60µs, tiempo apropiado para realizar la adquisición de la muestra. Después de capturada la muestra análoga se inicia el proceso de convertirla en una palabra digital, cuando la conversión esta completa el resultado es una palabra de diez bits almacenada en los registros ADRESH y ADRESL. Puesto que el modulador utiliza una palabra PCM de ocho bits se debe hacer un arreglo al resultado de diez bits, por esta razón se llama a la subrutina Divix4, esta se encarga de dividir por cuatro el resultado de diez bits para así obtener una palabra de ocho bits. Finalmente después de obtener la palabra de ocho bits la cual corresponde a la muestra análoga, se transmite de manera sincrónica con la ayuda de la USART. Para identificar el origen de la muestra se adiciona un noveno bit a la palabra de ocho bits, de esta manera si el noveno bit es cero indicara que la muestra proviene del canal cero, pero si es uno el origen es el canal uno. La transmisión de las muestras se hace de manera intercalada, es decir, primero se transmite la muestra del canal cero y luego la del canal uno. Para generar la señal PAM de una de las dos entradas análogas, se envía al puerto B la palabra de ocho bits de la muestra para que un conversor digital-análogo se encargue de construirla, un selector ubicado en el pin cero del puerto D decide cual de las dos entradas análogas generara la señal PAM.

Por ultimo se ejecuta un POP para restaurar el registro W y las banderas de Carry, Digit Carry y Cero que se habían almacenado a su registro original (STATUS).

8.13.1.3 Subrutina Divix4. Esta subrutina divide por cuatro el resultado de la conversión de la muestra análoga, este resultado es una palabra digital de diez bits la cual se almacena en dos registros de ocho bits, estos registros son ADRESH y ADRESL, el resultado de la división es una palabra de ocho bits que se almacena en el registro RESULTADO. Para realizar la división por cuatro se realizan dos desplazamientos a la derecha de los registros ADRESH y ADRESL.

8.13.2 Diagrama de flujo del transmisor. A continuación se presenta el diagrama de flujo de las rutinas de servicio a la interrupción y de la subrutina que fue necesario implementar en el desarrollo de la etapa de transmisión para el módulo PCM.

RUTINAS DE SERVICIO A LAS INTERRUPCIONES DEL TRANSMISOR PCM



8.13.3 Programa en código fuente del transmisor. A continuación se da el programa detallado, que se utilizo en el microcontrolador, con sus respectivos comentarios.

```
;-------------------------------------------------------------------------------------------------------------------
;                                        Modulador PCM (Transmisor)
;-------------------------------------------------------------------------------------------------------------------
;       PCMTX.ASM

        list    p=16F874        ;Procesador
        include <p16F874.inc>

;       Al programar el micro tener en cuenta los fusibles de configuración
;       OSC = HS, WDT = OFF, PWRTE = OFF, CP = OFF, BODEN = OFF
;       LVP = Disabled, CPD = OFF, WRT = Enabled, DEBUG = Disabled

;------------------------------------------------- Zona de Registros ---------------------------------------------

tmr_opt         equ     01h     ;
status          equ     03h     ;
puertob         equ     06h     ;
puertod         equ     08h     ;
intcon          equ     0bh     ;
pir_pie1        equ     0ch     ;
tmr2            equ     11h     ;
t2con_pr2       equ     12h     ;
adreshl         equ     1eh     ;
adcon0_1        equ     1fh     ;
rc_txsta        equ     18h     ;
txreg_brg       equ     19h     ;Registro de Transmisión / Generador de Baudios
delay           equ     20h     ;
resultado       equ     21h     ;
wtemp           equ     22h     ;Temporal w
stemp           equ     23h     ;Temporal STATUS
tempwr          equ     24h     ;Temporal del reloj

;------------------------------------------------- Asignaciones de bit -------------------------------------------

rp0             equ     5       ;Registro STATUS
go              equ     2       ;Registro ADCON0
adon            equ     0       ;Activa el conversor ADC
gie             equ     7       ;Habilita int. general
peie            equ     6       ;Habilita interrupciones periféricas
toie            equ     5       ;Habilita int. por tmr0
toif            equ     2       ;Flag de int. por tmr0
tmr2on          equ     2       ;Bit de activación (On) del tmr2
adif            equ     6       ;Registro PIR1
tmr2if          equ     1       ;Flag de int. por tmr2
tmr2ie          equ     1       ;Habilita int. por tmr2
csrc            equ     7       ;Selecciona fuente de reloj
txen            equ     5       ;Habilita transmisión
tx9             equ     6       ;Habilita transmisión de dato de 9 bits
sync            equ     4       ;Selección de modo USART
tx9d            equ     0       ;Noveno bit del dato transmitido
spen            equ     7       ;Habilita puerto serial
select          equ     0       ;Selector de señal PAM (señal1 / señal2)
```

```
;---------------------------------------------------- Definiciones ----------------------------------------------------

#define         banco0          bcf     status,rp0
#define         banco1          bsf     status,rp0
#define         on_ad           bsf     adcon0_1,adon
#define         off_ad          bcf     adcon0_1,adon
#define         go_ad           bsf     adcon0_1,go
#define         clrflag         bcf     pir_pie1,adif
#define         ontmr2          bsf     t2con_pr2,tmr2on
#define         offtmr2         bcf     t2con_pr2,tmr2on

;------------------------------------------------ Configuración del PIC ------------------------------------------------

        org     0
        goto    inicio          ;Inicio del programa principal

        org     4
        goto    inter           ;Vector de interrupciones

inicio  banco1                  ;Va al banco1

        movlw   04h             ;Configuración del tmr0, prescaler 1:32
        movwf   tmr_opt         ;Reloj interno
        movlw   066h            ;Carga del registro pr2 para el tmr2
        movwf   t2con_pr2       ;66h -> 332useg (aproximadamente)
        bsf     pir_pie1,tmr2ie ;Habilita la int. del tmr2 cuando pr2 = tmr2
        movlw   00h             ;Configuración de puertos
        movwf   puertob         ;Puertob como salida
        movlw   01h             ;Bit 0 del puertod como entrada
        movwf   puertod         ;
        movlw   080h            ;Config. ADC, Justificado der., Ref. (+Vdd,Vss)
        movwf   adcon0_1        ;
        movlw   d'255'          ;Configuración de la USART, a 19,53 Kbaudios
        movwf   txreg_brg       ;Carga del registro generador de baudios
        bsf     rc_txsta,csrc   ;Modo maestro
        bsf     rc_txsta,txen   ;Transmisión habilitada
        bsf     rc_txsta,sync   ;Configuración USART como modo sincrónico
        bsf     rc_txsta,tx9    ;9 bits

        banco0                  ;Va al banco0

        movlw   02h             ;Configuración del tmr2, prescaler 16
        movwf   t2con_pr2       ;
        bsf     rc_txsta,spen   ;Habilitación del puerto serial
        bsf     intcon,gie      ;
        bsf     intcon,toie     ;
        bsf     intcon,peie     ;

        clrf    tmr_opt         ;Inicialización
        clrf    tmr2            ;
        clrf    delay           ;
        clrf    resultado       ;
        clrf    puertob         ;
        clrf    puertod         ;

;------------------------------------------------------- Loop --------------------------------------------------------

loop    goto    loop            ;Programa principal, loop infinito
```

```
;---------------------------- RUTINA DE SERVICIO A LA INT. DEL TMRO Y TMR2 ----------------------

inter   movwf  wtemp          ;Push
        swapf  status,w       ;
        movwf  stemp          ;

        btfss  intcon,toif    ;Se verifica quien produjo la interrupción
        goto   time2          ;

;------------------------------------------------------------ Tmr0 ------------------------------------------------------------

time1   movf   tmr_opt,w      ;Ajuste de tiempo del timer0
        movwf  tempwr         ;
        btfss  tempwr,0       ;
        goto   time64         ;

time64  movlw  064h           ;64h -> 1mseg (aproximadamente)
        movwf  tmr_opt        ;Carga del timer0
        ontmr2                ;Se enciende el tmr2

;---------------------------------------------------- Muestreo del canal 0 ----------------------------------------------------

adc0    movlw  80h            ;Canal 0, Fosc/32
        movwf  adcon0_1       ;
sample  clrflag               ;Toma de la muestra de la señal 1
        on_ad                 ;Inicio de la adquisición
        movlw  064h           ;
        call   retardo        ;Tiempo para realizar la captura
        go_ad                 ;Inicio de la conversión
wait    nop                   ;
        btfss  pir_pie1,adif  ;Verifica fin de conversión
        goto   wait           ;
        off_ad                ;Se desactiva el conversor ADC
        clrflag               ;Clarea la bandera del conversor
        call   divix4         ;Arreglo de la muestra de 10 bits
        movf   resultado,w    ;
        btfss  puertod,select ;Si select = 0 -> sale dato por puertob (PAM)
        movwf  puertob        ;Salida de la muestra por el puertob
        banco1                ;Va al banco1
        bcf    rc_txsta,tx9d  ;Carga del bit 9 (0 indica muestra del canal0)
        banco0                ;Va al banco0
        movwf  txreg_brg      ;Transmisión del dato

;---------------------------------------------------- Muestreo del canal 1 ----------------------------------------------------

adc1    movlw  88h            ;Canal 1, Fosc/32
        movwf  adcon0_1       ;
sampl1  clrflag               ;Toma de la muestra de la señal 2
        on_ad                 ;Inicio de la adquisición
        movlw  064h           ;
        call   retardo        ;Tiempo para realizar la captura
        go_ad                 ;Inicio de la conversión
wait1   nop                   ;
        btfss  pir_pie1,adif  ;Verifica fin de conversión
        goto   wait1          ;
        off_ad                ;Se desactiva el conversor ADC
        clrflag               ;Clarea la bandera del conversor
        call   divix4         ;Arreglo de la muestra de 10 bits
        movf   resultado,w    ;
        btfsc  puertod,select ;Si select = 1 -> sale dato por puertob (PAM)
```

```
        movwf  puertob          ;Salida de la muestra por el puertob
        banco1                  ;Va al banco1
        bsf    rc_txsta,tx9d    ;Carga del bit 9 (1 indica muestra del canal1)
        banco0                  ;Retorno al banco0
        movwf  txreg_brg        ;Transmisión del dato

        bcf    intcon,toif      ;Se clarea la bandera del tmr0
        goto   pop              ;

;------------------------------------------------------------ Tmr2 ----------------------------------------------------------

time2   offtmr2                 ;Se desactiva el timer2
        clrf   puertob          ;Se Clarea el puertob para generar una señal
        clrf   tmr2             ;PAM con un ciclo de trabajo (d) = 1/3
        bcf    pir_pie1,tmr2if  ;Se clarea la bandera del tmr2


pop     swapf  stemp,w          ;POP
        movwf  status           ;
        swapf  wtemp,f          ;
        swapf  wtemp,w          ;
        retfie                  ;

;------------------------------------------------- Subrutina retardo ------------------------------------------------------

retardo movwf  delay            ;Rutina de tiempo utilizada por el ADC.
dec     decfsz delay,f          ;
        goto   dec              ;
        return                  ;

;------------------------------------------------- Subrutina divix4 -------------------------------------------------------

divix4  bcf    status,0         ;División de la palabra de 10 bits por 4
        btfsc  adreshl,0        ;para convertirla a una palabra de 8 bits.
        bsf    status,0         ;
        banco1                  ;
        rrf    adreshl,f        ;Rota primera vez a la derecha
        banco0                  ;
        bcf    status,0         ;
        btfsc  adreshl,1        ;
        bsf    status,0         ;
        banco1                  ;
        rrf    adreshl,f        ;Rota segunda vez a la derecha
        movf   adreshl,w        ;
        banco0                  ;
        movwf  resultado        ;Palabra de 8 bits = Resultado
        return                  ;

;-------------------------------------------------------------------------------------------------------------------------
        end
```

8.13.4 Software del receptor. El software del receptor es el encargado de demodular la señal PCM y recuperar la información análoga.

El programa principal revisa el estado del registro de recepción, si este se llena quiere decir que se recibió un dato, luego se verifica el estado lógico del noveno bit de la palabra recibida, este bit identifica el origen de la muestra, si este coincide con el estado de un selector ubicado en el pin cero del puerto D quiere decir que la muestra debe ser enviada al puerto B para que un conversor digital-análogo reconstruya la señal análoga.

A continuación se hace una descripción detallada de la rutina implementada para el desarrollo del demodulador PCM.

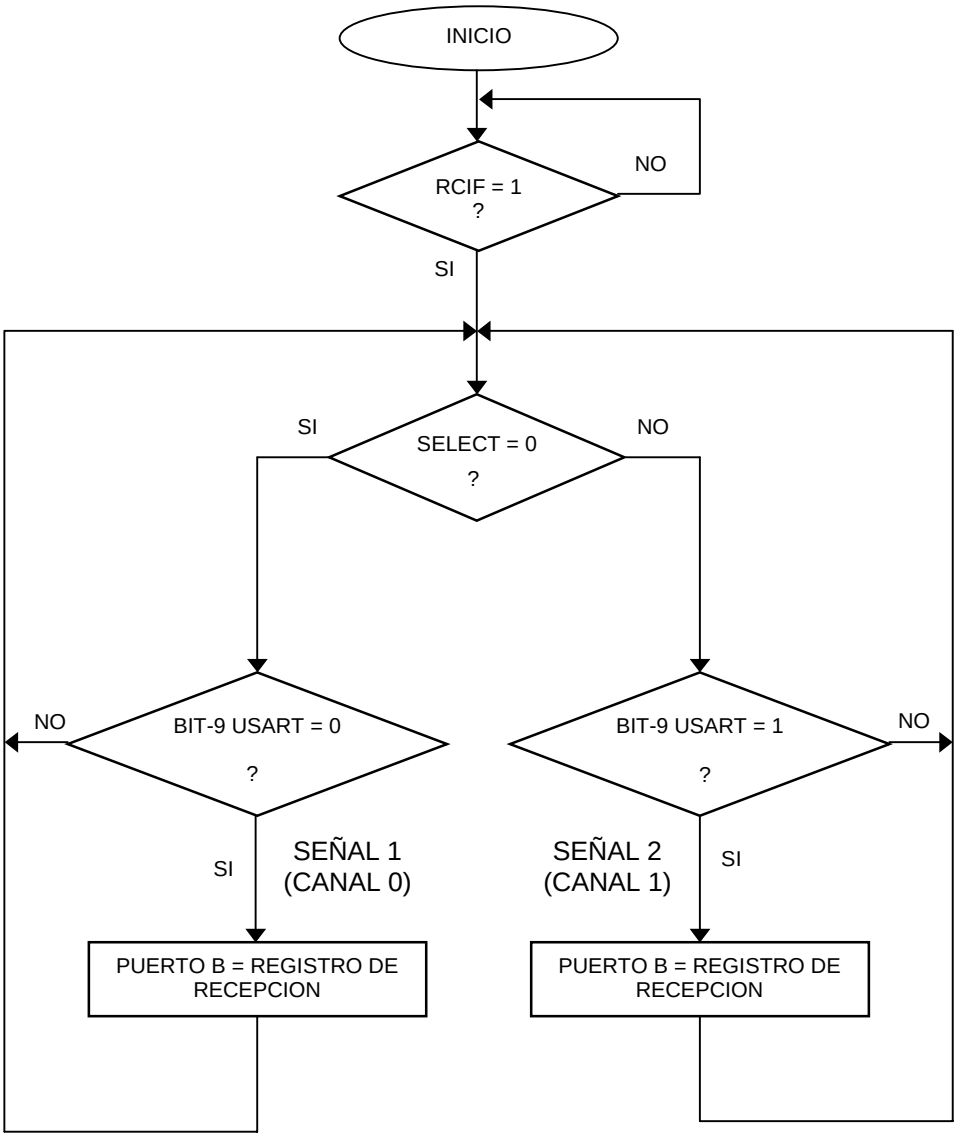
8.13.4.1 Programa principal. Al inicio de esta subrutina se programa el puerto B como salida y el bit cero del puerto D como entrada, se configura la USART a 19,53Kbaudios, en modo esclavo y sincrónico, utilizando 9 bits de datos, se habilita la transmisión y se habilita el puerto serial, por ultimo se inicializan todos los registros que se utilizan en el programa. Esto es realizado únicamente al inicio del programa y no se vuelve a entrar en el mientras el programa este corriendo.

Después la rutina revisa el registro de recepción utilizando la técnica de *polling*, es decir, se esta revisando continuamente la bandera RCIF para determinar si ha ocurrido un cambio de estado, si RCIF es uno indica que el registro de

recepción esta lleno, de esta manera el dato recibido es leído para luego verificar el estado lógico del noveno bit. Si el noveno bit es cero indicara que la muestra proviene del canal cero (SEÑAL 1), pero si es uno la muestra proviene del canal uno (SEÑAL 2). Con el fin de reconstruir solo una de las dos señales análogas se utiliza un selector ubicado en el pin cero del puerto D, si el estado lógico de este selector es cero indicara que se deben enviar al puerto B las muestras que se reciban del canal cero, pero si es uno las muestras enviadas al puerto B serán las del canal uno. A la salida del puerto B hay un conversor digital-análogo que se encarga de reconstruir la señal análoga. Luego el programa retorna a verificar de nuevo el estado de la bandera RCIF, es decir, a esperar que llegue un nuevo dato.

8.13.5 Diagrama de flujo del receptor. A continuación se presenta el diagrama de flujo del programa principal que fue necesario implementar en el desarrollo de la etapa de recepción para el módulo PCM.

PROGRAMA PRINCIPAL DEL RECEPTOR PCM



8.13.6 Programa en código fuente del receptor. A continuación se da el programa detallado, que se utilizo en el microcontrolador, con sus respectivos comentarios.

```
;-------------------------------------------------------------------------------------------------------------------
;                                        Demodulador PCM (Receptor)
;-------------------------------------------------------------------------------------------------------------------
;       PCMRX.ASM

        list    p=16F874        ;Procesador
        include <p16F874.inc>

;       Al programar el micro tener en cuenta los fusibles de configuración
;       OSC = HS, WDT = OFF, PWRTE = OFF, CP = OFF, BODEN = OFF
;       LVP = Disabled, CPD = OFF, WRT = Enabled, DEBUG = Disabled

;------------------------------------------------- Zona de Registros ---------------------------------------------

tmr_opt         equ     01h     ;
status          equ     03h     ;
puertob         equ     06h     ;
puertod         equ     08h     ;
intcon          equ     0bh     ;
pir_pie1        equ     0ch     ;
rc_txsta        equ     18h     ;
txreg_brg       equ     19h     ;Registro de Transmisión / Generador de Baudios
rcreg           equ     1ah     ;Registro de recepción


;------------------------------------------------ Asignaciones de bit ------------------------------------------

rp0             equ     5       ;Registro STATUS
gie             equ     7       ;Habilita int. general
peie            equ     6       ;Habilita interrupciones periféricas
rcif            equ     5       ;Bandera de int. por recepción
csrc            equ     7       ;Selecciona fuente de reloj
txen            equ     5       ;Habilita transmisión
rx9             equ     6       ;Habilita recepción de dato de 9 bits
sync            equ     4       ;Selección de modo USART
rx9d            equ     0       ;Noveno bit del dato recibido
spen            equ     7       ;Habilita puerto serial
sren            equ     5       ;Habilita recepción simple
cren            equ     4       ;Habilita recepción continua
select          equ     0       ;Selector de señal de salida


;------------------------------------------------- Definiciones ------------------------------------------------


#define         banco0          bcf     status,rp0
#define         banco1          bsf     status,rp0
```

```
;------------------------------------------------ Configuración del PIC ------------------------------------------------

        org     0               ;Inicio del programa principal

inicio  banco1                  ;Va al banco1

        movlw   00h             ;Configuración de puertos
        movwf   puertob         ;Puertob como salida
        movlw   01h             ;Bit0 del puertod como entrada
        movwf   puertod         ;

        movlw   d'255'          ;Configuración de la USART a 19.53 Kbaudios
        movwf   txreg_brg       ;Carga del registro generador de baudios
        bcf     rc_txsta,csrc   ;Modo esclavo
        bsf     rc_txsta,sync   ;Configuración USART como modo sincrónico

        banco0                  ;Retorno al banco0

        bsf     rc_txsta,spen   ;Habilitación del puerto serial
        bsf     rc_txsta,rx9    ;9 bits
        bcf     rc_txsta,sren   ;Deshabilita recepción simple
        bsf     rc_txsta,cren   ;Habilita recepción continua

        bsf     intcon,gie      ;
        bsf     intcon,peie     ;
        bcf     pir_pie1,rcif   ;

        clrf    puertob         ;Inicialización
        clrf    puertod         ;

;------------------------------------------------ PROGRAMA PRINCIPAL ------------------------------------------------

recibir btfss   pir_pie1,rcif   ;Polling, revisión del registro de recepción
        goto    recibir         ;

        btfsc   puertod,select  ;Selección de señal a salir por el puertob
        goto    canal1          ;

canal0  btfsc   rc_txsta,rx9d   ;Origen de la muestra = canal0
        goto    retorno         ;
        movf    rcreg,w         ;
        movwf   puertob         ;Se envía el dato recibido al puertob
        goto    retorno         ;

canal1  btfss   rc_txsta,rx9d   ;Origen de la muestra = canal1
        goto    retorno         ;
        movf    rcreg,w         ;
        movwf   puertob         ;Se envía el dato recibido al puertob

retorno bcf     pir_pie1,rcif   ;Clareo de la bandera de recepción
        movf    rcreg,w         ;
        goto    recibir         ;

;-------------------------------------------------------------------------------------------------------------------------
        end
```

8.14 HARDWARE DEL MODULO PCM

8.14.1 Funcionamiento del transmisor. El diagrama esquemático de la Figura 43 muestra dos generadores de funciones XR2206 que son los encargados de generar la entrada análoga del sistema PCM, estos generadores permiten elegir entre dos tipos de señal (sinusoidal o triangular) por medio de un selector (*dip switch)*, también se puede ajustar la frecuencia y la amplitud de las señales con la ayuda de un par de potenciómetros (*Trimmer*).

Las dos señales análogas son enviadas a los pines 2 (RA0/AN0) y 3 (RA1/AN1) del microcontrolador PIC16F874, estas son las entradas análogas del conversor análogo-digital, ambas están limitadas a un máximo de +5V por la acción de un diodo zener.

El microcontrolador se encarga de procesar la información análoga, es decir, toma una muestra de las señales, realiza una multiplexación de las muestras (TDM) y genera la señal PCM correspondiente a cada una de ellas, esta señal se obtiene en el pin 26 (RC7/RX/DT) del PIC. La señal de salida del microcontrolador es del tipo *unipola*r *NRZ* y esta es enviada con ayuda de la USART incorporada, a la línea entre el transmisor y el receptor por el pin 3 del conector DB9 de salida.

Un selector (*dip switch*) ubicado en el pin 19 (RD0) del PIC16F874 se encarga de conmutar la señal PAM que se visualizara y que pertenece a una de las dos señales análogas provenientes de los generadores de funciones. Para generar

la señal PAM se usa un conversor digital-análogo DAC0832 que se encarga de convertir a un voltaje análogo la muestra que captura el conversor análogo-digital interno del PIC y que se obtiene por el puerto B (RB0 a RB7) del mismo.

A la salida del DAC se utiliza un circuito integrado LF353 con dos amplificadores operacionales, uno de los cuales se utiliza como conversor de Corriente-Voltaje, ya que la salida del DAC es en forma de corriente; y el otro como inversor de voltaje, esto debido a que el primer operacional invierte la salida.

El transmisor tiene además cuatro puntos de prueba donde se puede observar con la ayuda de un Osciloscopio las formas de onda asociadas al proceso de modulación, estos puntos son los siguientes:

- CLK: El cual tiene la señal de sincronismo que utiliza la USART del PIC para realizar una transmisión sincrónica.
- DATO: En este punto se observa la trama TDM-PCM presente en la línea entre el módulo transmisor y el receptor.
- GND: Tierra (se envía al terminal común del Osciloscopio).
- PAM: Punto donde se observa la señal PAM que corresponde a una de las dos señales análogas de entrada (SEÑAL1/SEÑAL2).
- Las entradas análogas se obtienen de los pines 2 (SEÑAL1) y 3 (SEÑAL 2) del microcontrolador PIC16F874.

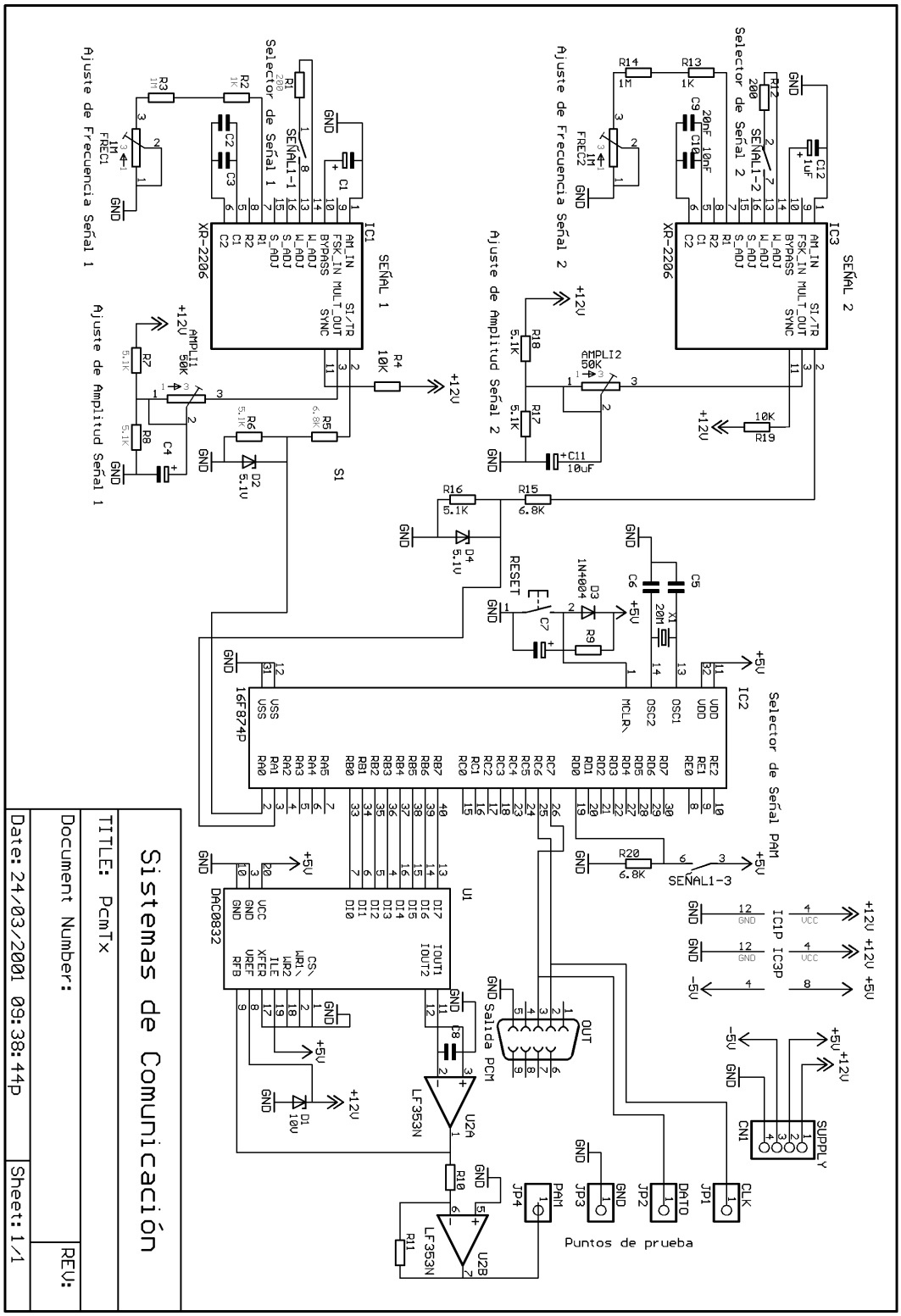
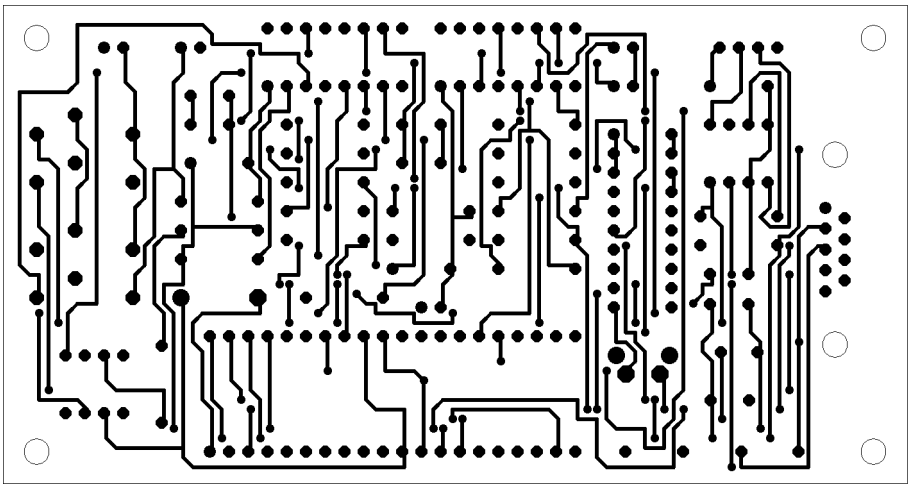
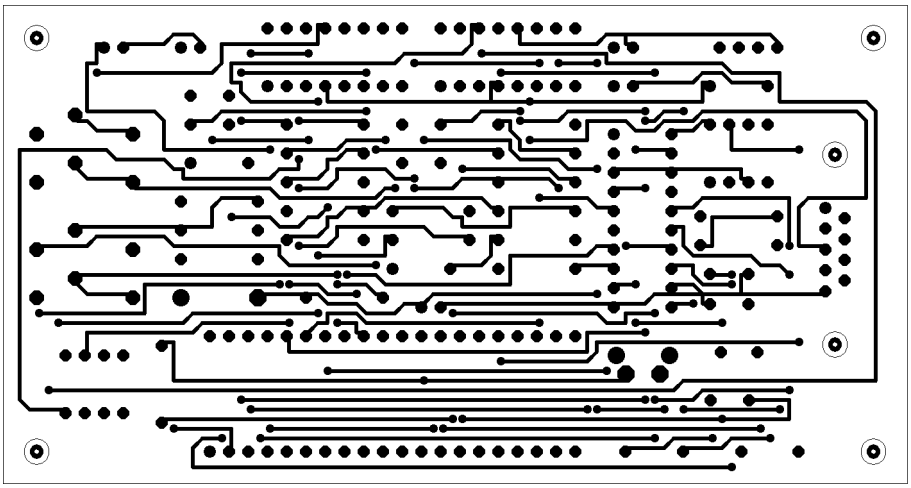


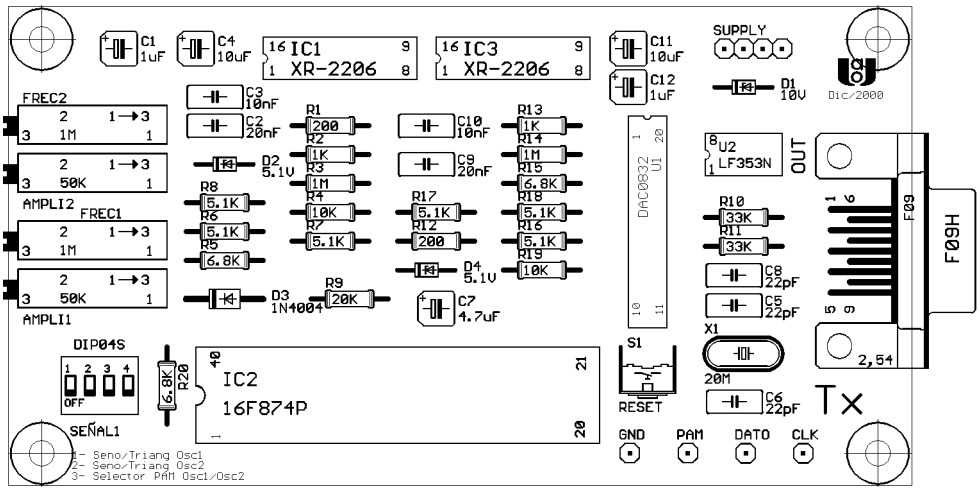
Figura 43 *Diagrama esquemático del transmisor PCM*



(a) Vista Superior



(b) Vista Inferior



(c) Vista de componentes

Figura 44 *Circuito Impreso del transmisor PCM*

8.14.2 Funcionamiento del receptor. Se puede ver en el diagrama de la Figura 45, que la señal PCM multiplexada de tipo *unipolar NRZ* llega al pin 26 (RC7/RX/DT) del microcontrolador por medio del pin 3 del conector DB9 de entrada.

El PIC16F874 procesa la información serial recibida y genera una salida en paralelo por el puerto B (RB0 a RB7), los datos en paralelo enseguida se convierten en una aproximación de la muestra analógica por medio del conversor digital-análogo DAC0832. Para elegir cual de las dos señales multiplexadas será recuperada, se conmuta la salida del puerto B del PIC con la ayuda de un selector (*dip switch*) ubicado en el pin 19 (RD0) del mismo. A la salida del DAC se utiliza un circuito integrado LF353 con dos amplificadores operacionales, uno de los cuales se utiliza como conversor de Corriente-Voltaje, ya que la salida del DAC es en forma de corriente; y el otro como inversor de voltaje, esto debido a que el primer operacional invierte la salida.
El receptor PCM posee cuatro puntos de prueba en los que se pueden ver con la ayuda de un Osciloscopio las formas de onda asociadas con el proceso de decodificación PCM, estos puntos son los siguientes:

- CLK: El cual tiene la señal de sincronismo que utiliza la USART del PIC para realizar una transmisión sincrónica.
- DATO: En este punto se observa la trama TDM-PCM de entrada.
- GND: Tierra (se envía al terminal común del Osciloscopio).
- SEÑALOUT: Señal recuperada (aproximación).

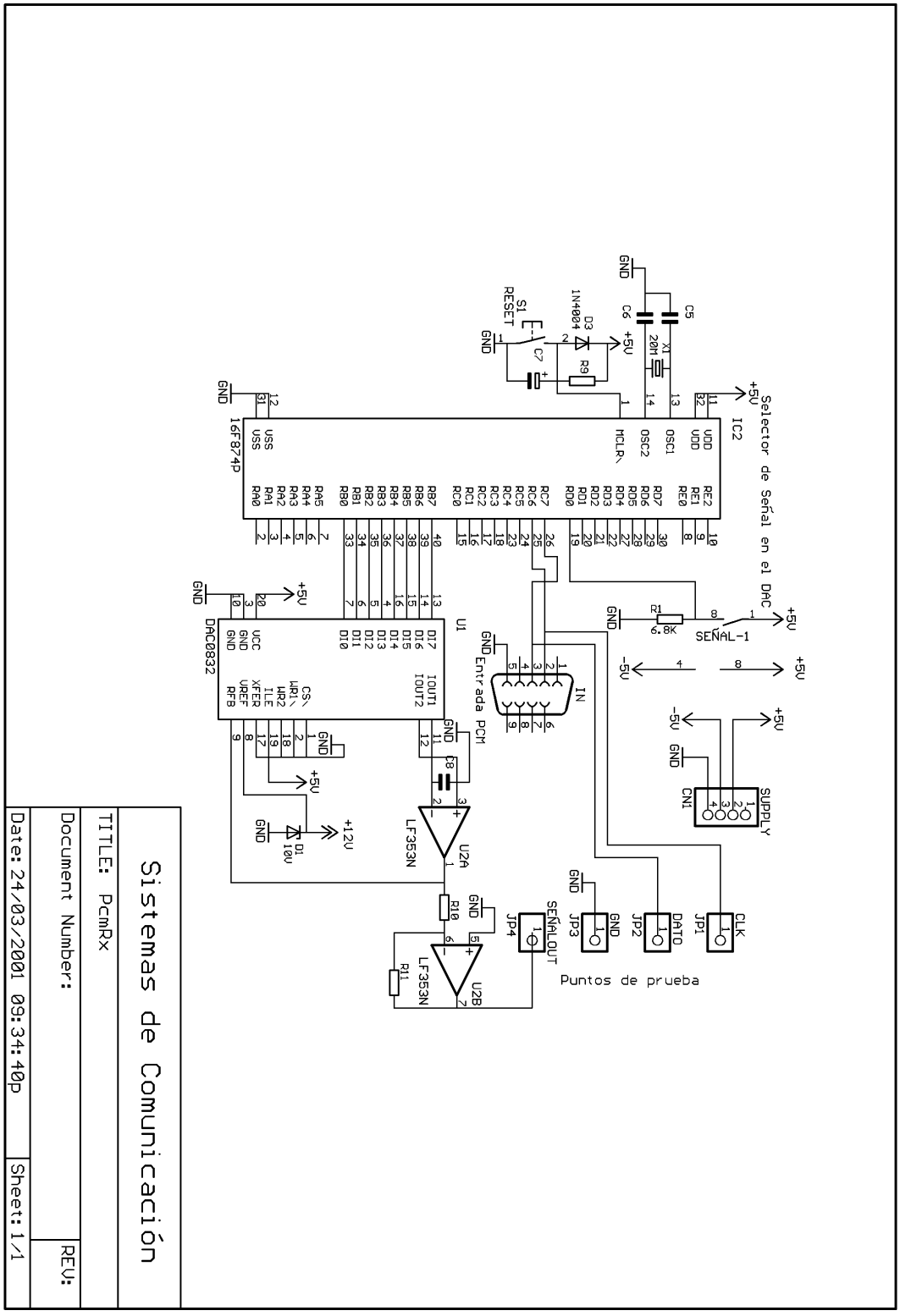
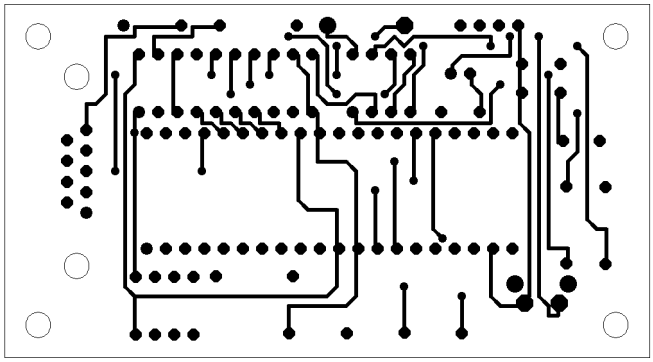
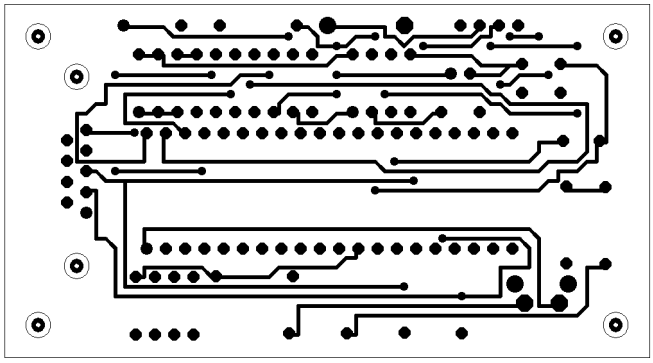


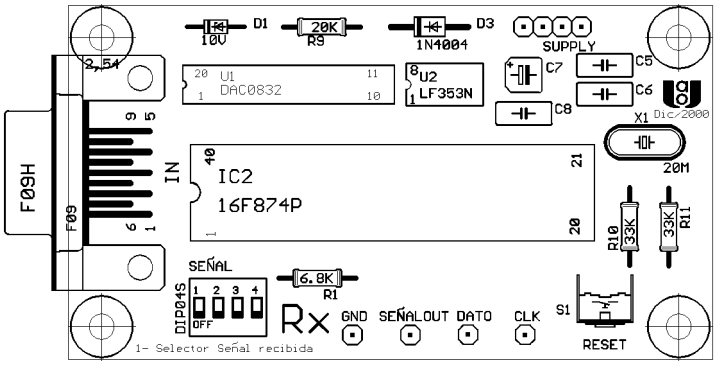
Figura 45 *Diagrama esquemático del receptor PCM*



(a) Vista Superior



(b) Vista Inferior



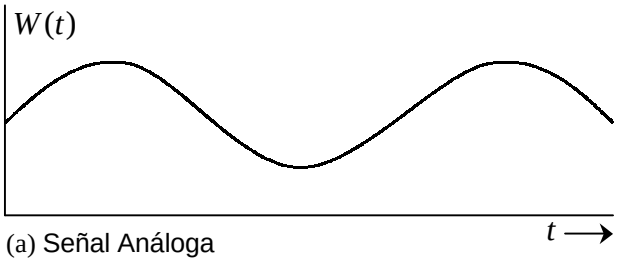
(c) Vista de componentes

Figura 46 *Circuito Impreso del receptor PCM*

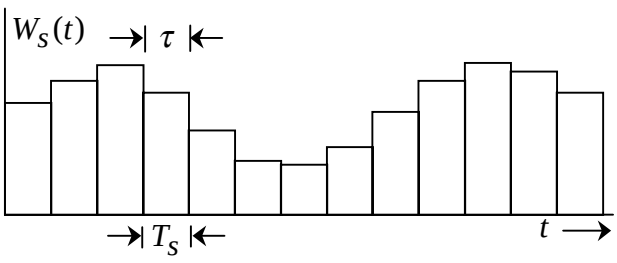
8.15 MODULO DELTA

Al igual que en el sistema PCM, el primer paso en la conversión de una señal análoga a una señal DELTA (digital) es el proceso de conversión de análogo a PAM. Luego se realiza el proceso de cuantización que se encarga de asignar a cada muestra de la señal análoga (PAM) un entero binario de *n* bits, cuyo valor será la mejor aproximación de la muestra. Estas operaciones son realizadas por el conversor análogo-digital incorporado en el microcontrolador, el cual tiene una resolución de 10 bits.

Al ser la señal PAM parte del procesamiento interno del conversor análogo-digital no es posible observarla. De manera que para poder visualizarla se toma la muestra binaria tomada por el conversor durante la interrupción y se envía a un conversor digital-análogo el cual reconstruye la señal PAM, esta señal tiene un ciclo de trabajo *d* igual a uno, ya que la apertura τ es igual a T_s, es decir, se aprovecha el 100% del pulso.



(a) Señal Análoga

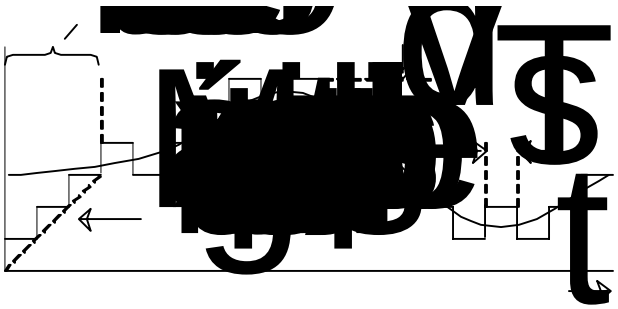


(b) Señal PAM Resultante (*d* = 1)

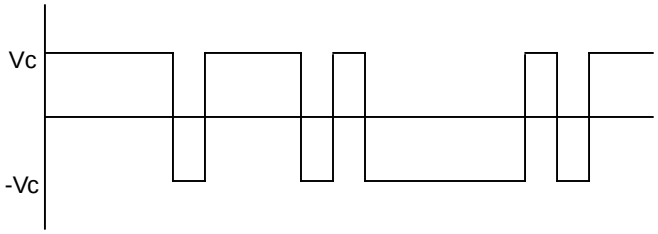
Figura 47 *Formas de onda de la señal Análoga y la señal PAM Resultante*

Se aprecia gráficamente un ciclo de trabajo igual a uno ya que no existe separación entre los pulsos, como se ve en la Figura 47.

Cada muestra tomada por el conversor análogo-digital es comparada con el contenido de un acumulador, de esta manera si el valor de la muestra es mayor que el del acumulador, el contenido de este ultimo se incrementara una unidad y además se enviara un uno lógico a la línea de transmisión, pero si el valor de la muestra es menor que el del acumulador, este se decrementara una unidad y se enviara un cero lógico.



a) Formas de onda de entrada análoga y de salida del acumulador.



b) Señal enviada por la línea de transmisión (1 bit).

Figura 48 Formas de onda del sistema DELTA

Se puede decir entonces que la salida del acumulador hace un rastreo de la entrada analógica, como se puede apreciar en la Figura 48. Pero la señal de salida del acumulador no siempre rastrea la señal de entrada análoga.

La señal errónea se produce por ruido de cuantización, este se clasifica en dos tipos: *ruido por sobrecarga de pendiente* y *ruido granular*. El ruido por sobrecarga de pendiente ocurre cuando el tamaño del escalón δv es demasiado pequeño como para que la salida del acumulador siga los rápidos cambios en la forma de la señal de entrada, es decir, tiene una pendiente que excede la máxima pendiente posible del acumulador.

El ruido granular ocurre con cualquier tamaño de escalón aunque es menor cuando el tamaño del escalón es pequeño y se presenta durante periodos de tiempo cuando no hay transición en la señal.

Teniendo en cuenta lo anterior se decide que tanto el conversor análogo-digital como el acumulador del módulo DELTA deben tener una resolución de siete bits, esto con el fin de permitir que la pendiente del acumulador pueda seguir la señal de la entrada análoga, ya que de esta manera se logra que el tamaño del escalón no sea tan pequeño. Para obtener esto se introduce el siguiente factor de conversión:

$$D_7 = \frac{D_{10}}{8}$$

donde D_7 es la palabra digital de siete bits, y D_{10} es la palabra digital de diez bits proveniente del conversor análogo-digital.

El factor de conversión modifica los niveles de cuantización del conversor como se aprecia a en la Figura 49.

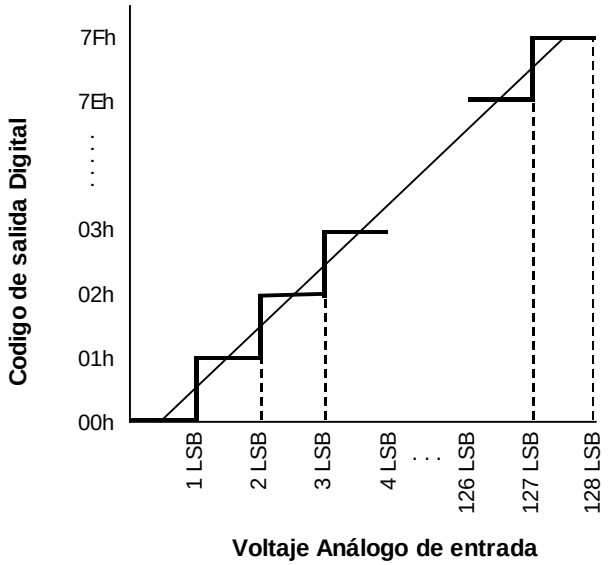


Figura 49 *Función de transferencia (Niveles de Cuantización) modificada a siete bits del conversor análogo-digital*

Para obtener el valor del tamaño del escalón δv del acumulador hallamos el valor de *1LSB* en el conversor análogo-digital, ya que estos dos son equivalentes, así:

$$\delta v = 1LSB = \frac{Vref}{2^N} = \frac{5}{2^7} = 39{,}06mV$$

donde *Vref* es el voltaje de referencia del conversor y *N* es el numero de bits de resolución. Usando la información anterior y teniendo en cuenta que para prevenir la sobrecarga de la pendiente se requiere que:

$$\left|\frac{df(t)}{t}\right|_{max} \leq \frac{\delta v}{T_s}$$

donde $|df(t)/t|_{máx.}$ representa la máxima pendiente de la señal análoga de entrada y δv es el tamaño del escalón.

Se obtiene el periodo de la frecuencia de muestreo para el sistema DELTA de la siguiente manera:

$$T_S \leq \frac{\delta v}{\left|\frac{df(t)}{t}\right|_{max}} = \frac{39{,}06mV}{350} = 111{,}6\mu s$$

para efectos de cálculo se asume que la señal análoga de entrada tendrá una máxima pendiente igual a 350, esta es una pendiente bastante pronunciada que no posee la señal que proviene del generador pero que se usa de referencia. Por lo tanto la mínima frecuencia de muestreo recomendable es:

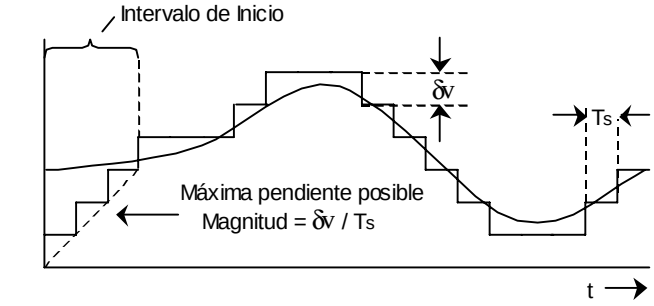
$$f_S = \frac{1}{T_S} = \frac{1}{111{,}6\mu s} = 8960mps$$

así se decide realizar un muestreo a razón de 10000mps (muestras por segundo), es decir, se tomara una muestra cada 100µs. Este tiempo es el que determina la interrupción que genera el Timer 0, encargado de capturar la muestra por medio del conversor análogo-digital.

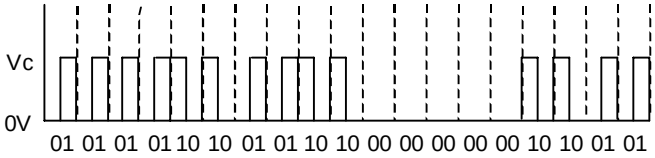
En el caso del ruido granular que aparece cuando la comparación entre los valores de la muestra actual y el acumulador dan como resultado una igualdad, haciendo que este ultimo oscile tratando de sostener un valor en el cual la señal análoga es constante, es minimizado haciendo que el mismo comparador permita al acumulador sostener el valor adecuado ante una igualdad.

La adición de una comparación más, es decir, se compara si la muestra es *menor, mayor* o *igual* al acumulador, hace necesario utilizar dos bits para determinar el resultado de la comparación, así cuando la muestra sea menor que el acumulador, a la salida del comparador habrá un cero (00_b), cuando sea igual habrá un uno (01_b) y cuando sea mayor habrá un dos (10_b).

Esta pequeña modificación hace que el acumulador realice un rastreo mucho más confiable de la entrada análoga, como se puede ver en la Figura 50.



a) Formas de onda de entrada análoga y de salida del acumulador al incluir la comparación de la igualdad.



b) Señal serial enviada por la línea de transmisión (2 bits).

Figura 50 *Nuevas Formas de onda del sistema DELTA*

Ya que finalmente lo que se envía por la línea de transmisión es la salida del comparador que consta de dos bits, estos deben ser enviados de forma serial, por lo que se utiliza la USART incorporada en el microcontrolador. El inconveniente de la USART es que utiliza una trama de transmisión de ocho bits y solo se necesita enviar dos bits, por lo que la velocidad de transmisión debe ser elegida de tal manera que la trama completa pueda ser transmitida en mucho menos de 100μs, es decir, antes de que la próxima interrupción se produzca, teniendo en cuenta que parte de ese tiempo lo utiliza el conversor análogo-digital para capturar la muestra y convertirla, lo que tarda aproximadamente 60μs, reduciendo el tiempo para realizar la transmisión a menos de 40μs, descontando también el tiempo de procesamiento de la muestra. Con esta restricción se decide utilizar una velocidad de transmisión de

294.1Kbaudios en modo sincrónico, ya que a esta velocidad la trama es enviada en un tiempo de 27,2µs, como se ve en la Figura 51.

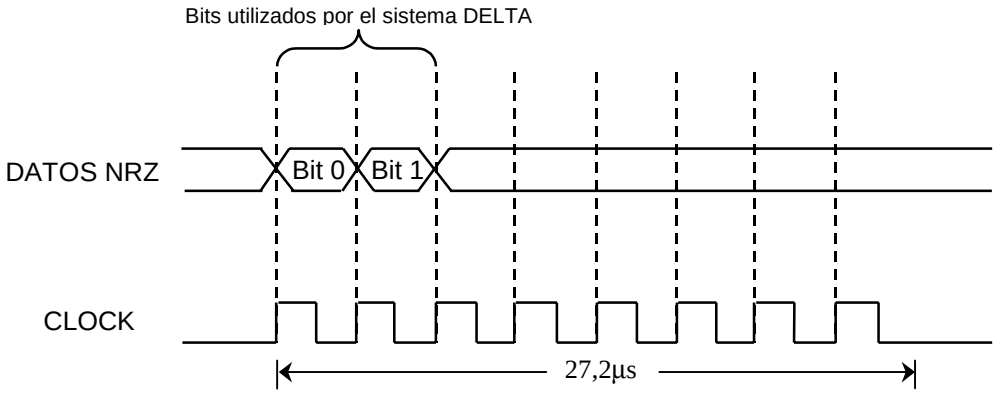


Figura 51 *Forma de onda Sincrónica DELTA*

En la etapa de recepción del sistema DELTA no se hacen comparaciones, solo existe un acumulador al cual se le realizan cambios de acuerdo a la información de la señal DELTA que se recibe, estos cambios en el acumulador son enviados a un conversor digital-análogo, recuperando así una aproximación de la señal análoga de entrada.

Resumiendo se puede describir el proceso de transmisión y recepción del módulo DELTA utilizando la Figura 52.

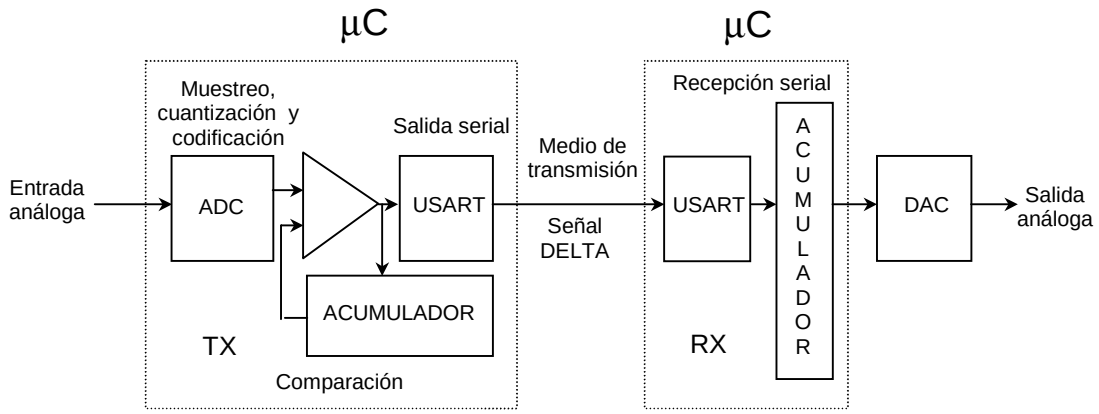


Figura 52 *Diagrama de bloques de los módulos de transmisión y recepción DELTA*

En la gráfica, el transmisor, realiza las operaciones de muestreo, cuantización y codificación de la señal análoga utilizando un conversor análogo-digital, luego la muestra codificada es comparada con el contenido de un acumulador, el resultado de la comparación determina si el mismo acumulador se debe incrementar, decrementar o dejar constante, el resultado del comparador también es transmitido serialmente utilizando una USART.

En la etapa de recepción, los datos llegan de nuevo a una USART que los convierte de serie a paralelo, el dato que es recibido, nuevamente determina si un acumulador debe decrementarse, incrementarse o permanecer constante, el contenido de este acumulador es enviado a un conversor digital-análogo que se encarga de reproducir una aproximación de la señal análoga.

8.15.1 Software del transmisor. El software del transmisor es quien realiza la modulación DELTA de la entrada análoga.

El programa principal es un loop infinito, el cual solo es interrumpido por el desbordamiento del Timer 0 que genera una interrupción. La interrupción del Timer 0 se produce cada 100µs. La interrupción dirige el programa a la dirección de memoria 0004H donde se encuentra el vector de interrupciones. En este sitio se ubica un salto que apunta a la dirección donde se encuentra la rutina de servicio a la interrupción. Esta rutina es la que finalmente se encarga del procesamiento de la entrada análoga y de la generación de las señales de salida.

La rutina de servicio a la interrupción toma una muestra de la entrada análoga utilizando el conversor análogo-digital, luego de obtener la palabra digital de la muestra esta se compara con el contenido de un acumulador, del resultado de esta comparación depende la palabra que será transmitida con la USART y la modificación que se haga al acumulador. La muestra tomada también se envía al puerto B, de esta manera un conversor digital-análogo construye la señal PAM.

A continuación se hace una descripción detallada de las rutinas implementadas para el desarrollo del modulador DELTA.

8.15.1.1 Programa principal. En esta parte del programa se configura el Timer 0, el cual utiliza la fuente de reloj interna del microcontrolador y el prescaler en 1:32, se programa el puerto B como salida, se configura el conversor análogo-digital con referencias de cinco voltios (+5V) y tierra, el resultado queda justificado a la derecha, se configura la USART a 294,1Kbaudios, en modo maestro y sincrónico, utilizando 8 bits de datos, se habilita la transmisión y se habilita el puerto serial, por ultimo se inicializan todos los registros que se utilizan en el programa y se habilita la interrupción del Timer 0. Esto es realizado únicamente al inicio del programa y no se vuelve a entrar en el mientras el programa este corriendo. Luego el programa ingresa a un loop infinito, es decir, se ejecuta un salto que apunta a la dirección de

memoria donde se encuentra el mismo salto, este lazo cerrado que se produce solo se interrumpe por el desbordamiento Timer 0 que genera una interrupción.

8.15.1.2 Rutina de servicio a la interrupción del Timer 0. Esta rutina inicialmente ejecuta un PUSH, el cual se encarga de almacenar el registro de trabajo W y las banderas de estado de la ALU como Carry, Digit Carry y Cero que se encuentran en el registro STATUS.

Luego se realiza el ajuste de tiempo del Timer 0, este ajuste se realiza para cargar el Timer 0 correctamente y de esta manera calibrar el tiempo en que se presentara la próxima interrupción. Realizado esto se procede a tomar la muestra de la entrada análoga, esto se realiza con la ayuda del conversor análogo-digital de ocho canales incorporado en el microcontrolador, de esta manera se toma una muestra del canal cero, el procedimiento para tomar la muestra se inicia con la elección del canal de entrada, después se activa el conversor análogo-digital, esto inicia la adquisición de la muestra para lo cual existe un tiempo requerido, este tiempo es contabilizado por la subrutina Retardo, esta subrutina ejecuta un retardo de aproximadamente 60μs, tiempo apropiado para realizar la adquisición de la muestra. Después de capturada la muestra análoga se inicia el proceso de convertirla en una palabra digital, cuando la conversión esta completa el resultado es una palabra de diez bits almacenada en los registros ADRESH y ADRESL. Puesto que el modulador utiliza una palabra DELTA de siete bits se debe hacer un arreglo al resultado de diez bits, por esta razón se llama a la subrutina Divix8, esta se encarga de

dividir por ocho el resultado de diez bits para así obtener una palabra de siete bits. Finalmente después de obtener la palabra de siete bits la cual corresponde a la muestra análoga, se realiza una comparación de la muestra con un registro acumulador, este acumulador es un contador ascendente y descendente, el resultado de la comparación determina la palabra que se transmitirá, es decir, si la muestra es menor que el acumulador se transmitirá un cero y se decrementa en uno el acumulador, si es igual se transmitirá un uno y el acumulador no sufre cambios y si es mayor se transmitirá un dos y se incrementa en uno el acumulador. La transmisión se hace con la ayuda de la USART también incorporada al microcontrolador.

Para generar la señal PAM de la entrada análoga, se envía al puerto B la palabra de siete bits de la muestra para que un conversor digital-análogo se encargue de construirla.

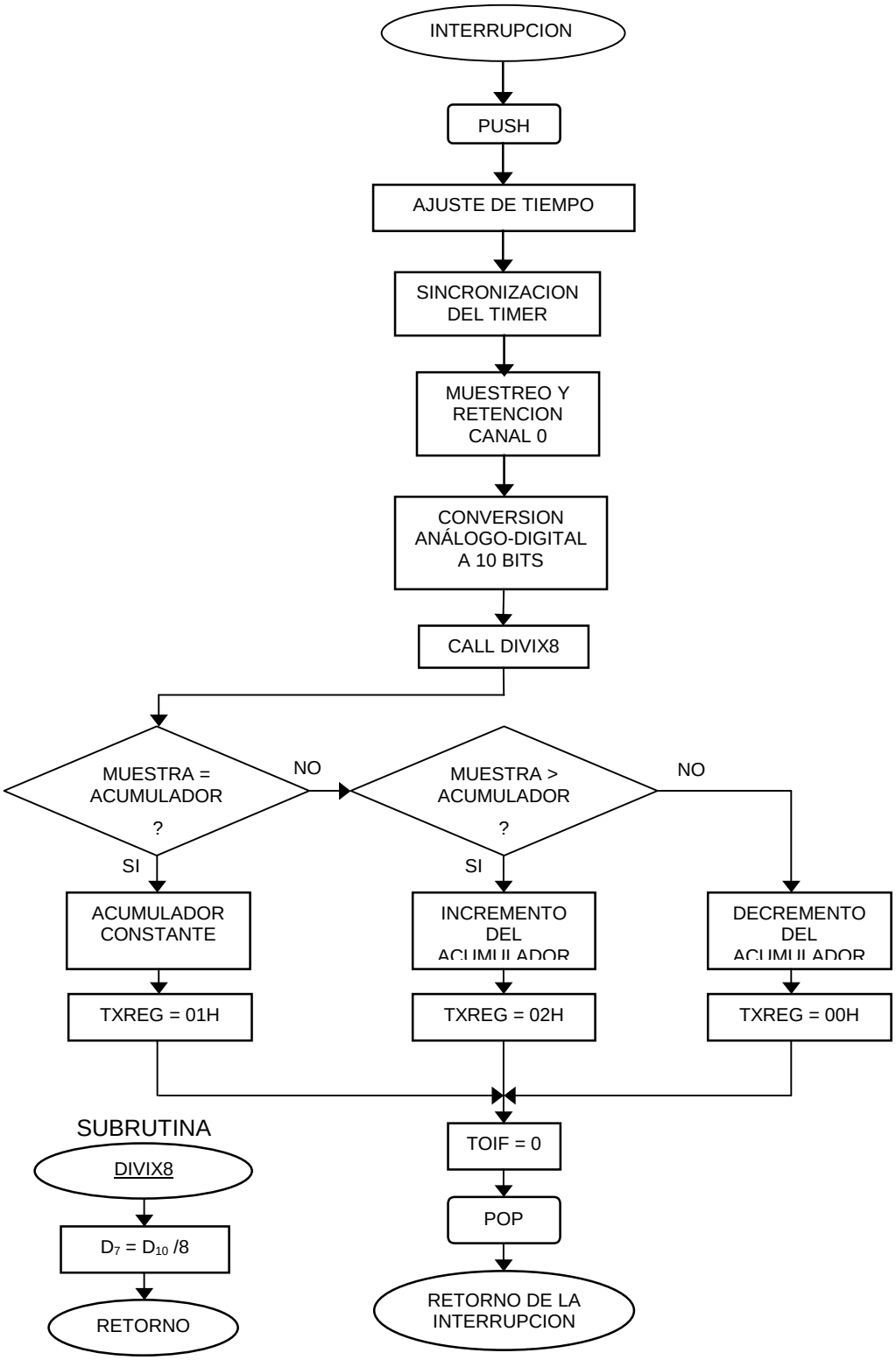
Por ultimo se ejecuta un POP para restaurar el registro W y las banderas de Carry, Digit Carry y Cero que se habían almacenado a su registro original (STATUS).

8.15.1.3 Subrutina Divix8. Esta subrutina divide por ocho el resultado de la conversión de la muestra análoga, este resultado es una palabra digital de diez bits la cual se almacena en dos registros de ocho bits, estos registros son ADRESH y ADRESL, el resultado de la división es una palabra de siete bits que se almacena en el registro RESULTADO. Para realizar la división por

cuatro se realizan tres desplazamientos a la derecha de los registros ADRESH y ADRESL.

8.15.2 Diagrama de flujo del transmisor. A continuación se presenta el diagrama de flujo de la rutina de servicio a la interrupción y de la subrutina que fue necesario implementar en el desarrollo de la etapa de transmisión para el módulo DELTA.

RUTINA DE SERVICIO A LA INTERRUPCION DEL TRANSMISOR DELTA



8.15.3 Programa en código fuente del transmisor. A continuación se da el programa detallado, que se utilizo en el microcontrolador, con sus respectivos comentarios.

```
;-------------------------------------------------------------------------------------------------------------------
;                                        Modulador DELTA (Transmisor)
;-------------------------------------------------------------------------------------------------------------------
;       DELTATX.ASM

        list    p=16F874        ;Procesador
        include <p16F874.inc>

;       Al programar el micro tener en cuenta los fusibles de configuración
;       OSC = HS, WDT = OFF, PWRTE = OFF, CP = OFF, BODEN = OFF
;       LVP = Disabled, CPD = OFF, WRT = Enabled, DEBUG = Disabled


;------------------------------------------------- Zona de Registros ----------------------------------------------


tmr_opt         equ     01h     ;
status          equ     03h     ;
puertob         equ     06h     ;
intcon          equ     0bh     ;
pir_pie1        equ     0ch     ;
adreshl         equ     1eh     ;
adcon0_1        equ     1fh     ;
rc_txsta        equ     18h     ;
txreg_brg       equ     19h     ;Registro de transmisión / Generador de Baudios.
delay           equ     20h     ;
resultado       equ     21h     ;
wtemp           equ     22h     ;Temporal w
stemp           equ     23h     ;Temporal STATUS
tempwr          equ     24h     ;Temporal del reloj
up_down         equ     25h     ;Acumulador ascendente / descendente


;------------------------------------------------- Asignaciones de bit ------------------------------------------


rp0             equ     5       ;Registro STATUS
go              equ     2       ;Registro ADCON0
adon            equ     0       ;Activa el conversor ADC
gie             equ     7       ;Habilita int. general
peie            equ     6       ;Habilita interrupciones periféricas
toie            equ     5       ;Habilita int. por tmr0
toif            equ     2       ;Flag de int. por tmr0
adif            equ     6       ;Registro PIR1
csrc            equ     7       ;Selecciona fuente de reloj
txen            equ     5       ;Habilita transmisión
tx9             equ     6       ;Habilita transmisión de dato de 9 bits
sync            equ     4       ;Selección de modo USART
spen            equ     7       ;Habilita puerto serial
```

```
;---------------------------------------------------- Definiciones ----------------------------------------------------

#define         banco0          bcf     status,rp0
#define         banco1          bsf     status,rp0
#define         on_ad           bsf     adcon0_1,adon
#define         off_ad          bcf     adcon0_1,adon
#define         go_ad           bsf     adcon0_1,go
#define         clrflag         bcf     pir_pie1,adif

;---------------------------------------------------- Configuración del PIC ----------------------------------------------------

        org     0
        goto    inicio          ;Inicio del programa principal

        org     4
        goto    inter           ;Vector de interrupciones

inicio  banco1                  ;Va al banco1

        movlw   04h             ;Configuración del tmr0, prescaler 1:32
        movwf   tmr_opt         ;Fuente de Clock interna
        movlw   00h             ;Configuración de puertos
        movwf   puertob         ;Puertob como salida
        movlw   080h            ;Config. ADC, Justificado der., Ref. (+Vdd,Vss)
        movwf   adcon0_1        ;
        movlw   d'16'           ;Configuración de la USART a 294.1 KBaudios
        movwf   txreg_brg       ;Carga del registro generador de baudios
        bsf     rc_txsta,csrc   ;Modo maestro
        bsf     rc_txsta,txen   ;Transmisión habilitada
        bsf     rc_txsta,sync   ;Configuración USART como modo sincrónico
        bcf     rc_txsta,tx9    ;8 bits

        banco0                  ;Retorno al banco0

        bsf     rc_txsta,spen   ;Habilitación del puerto serial
        bsf     intcon,gie      ;
        bsf     intcon,toie     ;
        bsf     intcon,peie     ;
        clrf    tmr_opt         ;Inicialización
        clrf    delay           ;
        clrf    resultado       ;
        clrf    puertob         ;
        clrf    up_down         ;

;---------------------------------------------------------- Loop ----------------------------------------------------------

loop    goto    loop            ;Programa principal, loop infinito


;------------------------- RUTINA DE SERVICIO A LA INTERRUPCION DEL TMR0 ----------------------

inter   movwf   wtemp           ;Push
        swapf   status,w        ;
        movwf   stemp           ;

        movf    tmr_opt,w       ;Ajuste de tiempo del timer0
        movwf   tempwr          ;
        btfss   tempwr,0        ;
        goto    timef1          ;
```

```
timef1  movlw  0f1h            ;f1h -> 100useg (aproximadamente)
        movwf  tmr_opt         ;Carga del Timer0

;---------------------------------------------------------- Muestreo ----------------------------------------------------------

adc0    movlw  80h             ;Canal 0, Fosc/32
        movwf  adcon0_1        ;
sample  clrflag                ;Toma de la muestra de la señal
        on_ad                  ;Inicio de la adquisición
        movlw  064h            ;
        call   retardo         ;Tiempo para realizar la captura
        go_ad                  ;Inicio de la conversión
wait    nop                    ;
        btfss  pir_pie1,adif   ;Verifica el fin de la conversión
        goto   wait            ;
        off_ad                 ;Se desactiva el conversor ADC
        clrflag                ;Clarea la bandera del conversor
        call   divix8          ;Arreglo de la muestra

;------------------------------------------------- Generación de PAM ----------------------------------------------------

        movf   resultado,w     ;La muestra de 7 bits es enviada al puertob
        movwf  puertob         ;para que un DAC reconstruya una señal PAM (d = 1)

;------------------------------------------------------ Comparación -----------------------------------------------------

        movf   up_down,w       ;Comparación de la muestra con el acumulador
        subwf  resultado,w     ;
        btfss  status,0        ;Menor?
        goto   down            ;

        btfsc  status,2        ;Igual?
        goto   igual           ;

up      incf   up_down,f       ;Muestra > Acumulador -> Incremento del acumulador
        movlw  02h             ;
        movwf  txreg_brg       ;Se transmite un 02h
        goto   salida          ;

igual   movlw  01h             ;Muestra = Acumulador -> Acumulador no cambia
        movwf  txreg_brg       ;Se transmite un 01h
        goto   salida          ;

down    decf   up_down,f       ;Muestra < Acumulador -> Decremento del acumulador
        movlw  00h             ;
        movwf  txreg_brg       ;Se transmite un 00h

salida  bcf    intcon,toif     ;Se clarea la bandera del tmr0
pop     swapf  stemp,w         ;Pop
        movwf  status          ;
        swapf  wtemp,f         ;
        swapf  wtemp,w         ;
        retfie                 ;

;------------------------------------------------- Subrutina retardo ----------------------------------------------------

retardo movwf  delay           ;Rutina de tiempo utilizada por el ADC.
dec     decfsz delay,f         ;
        goto   dec             ;
        return                 ;
```

```
;------------------------------------------------ Subrutina divix8 ------------------------------------------------

divix8  bcf     status,0        ;Division de la palabra de 10 bits por 8
        btfsc   adreshl,0       ;para convertirla a una palabra de 7 bits.
        bsf     status,0        ;
        banco1                  ;
        rrf     adreshl,f       ;Primera rotación a la derecha
        banco0                  ;
        bcf     status,0        ;
        btfsc   adreshl,1       ;
        bsf     status,0        ;
        banco1                  ;
        rrf     adreshl,f       ;Segunda rotación a la derecha
        bcf     status,0        ;
        rrf     adreshl,f       ;Tercera rotación a la derecha
        movf    adreshl,w       ;
        banco0                  ;

        movwf   resultado       ;Palabra de 7 bits = Resultado
        return                  ;

;-----------------------------------------------------------------------------------------------------------------------
        end
```

8.15.4 Software del receptor. El software del receptor es el encargado de demodular la señal DELTA y recuperar la información análoga.

El programa principal revisa el estado del registro de recepción, si este se llena quiere decir que se recibió un dato, luego se verifica si el dato recibido fue un cero, un uno o un dos, de acuerdo a esto se modifica un registro acumulador, el contenido de este acumulador se envía al puerto B para que un conversor análogo digital reconstruya la señal análoga.

A continuación se hace una descripción detallada de la rutina implementada para el desarrollo del demodulador DELTA.

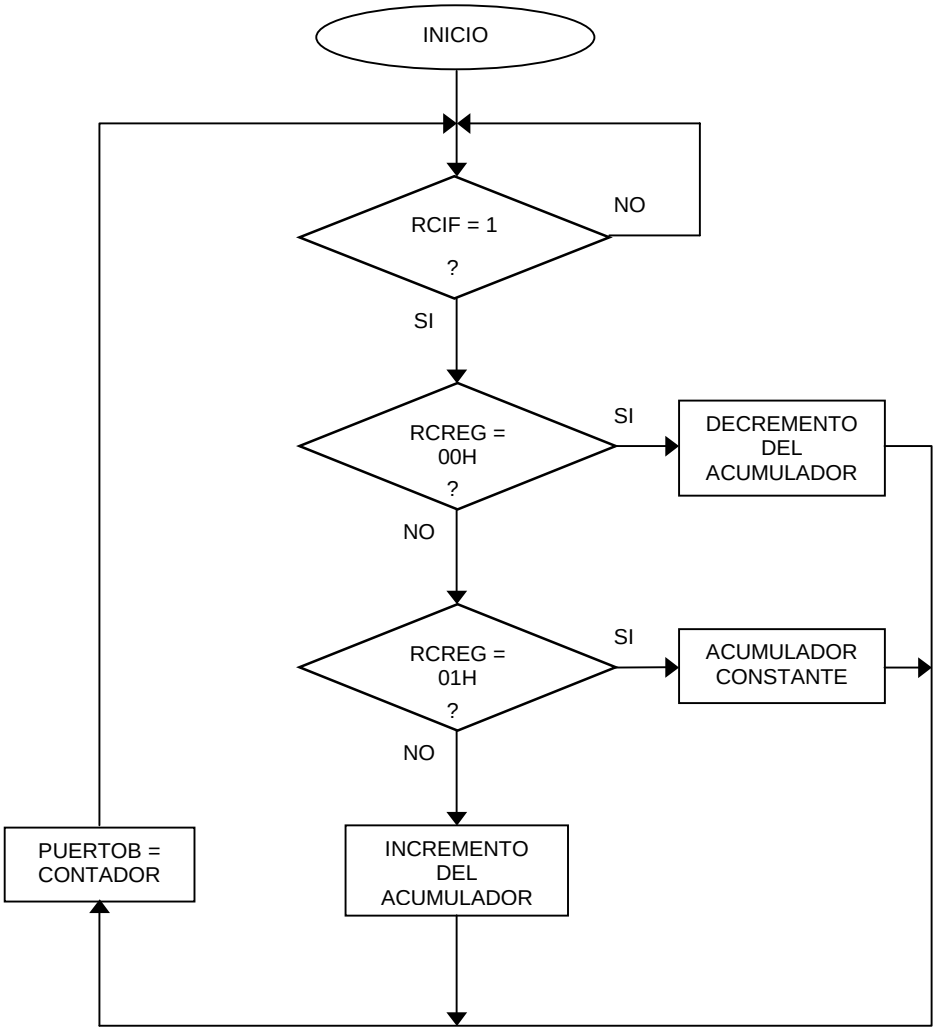
8.15.4.1 Programa principal. Al inicio de esta rutina se programa el puerto B como salida, se configura la USART a 294,1Kbaudios, en modo esclavo y sincrónico, utilizando 8 bits de datos, se habilita la transmisión y se habilita el puerto serial, por ultimo se inicializan todos los registros que se utilizan en el programa. Esto es realizado únicamente al inicio del programa y no se vuelve a entrar en el mientras el programa este corriendo.

Luego esta rutina revisa el registro de recepción utilizando la técnica de *polling*, es decir, se esta revisando continuamente la bandera RCIF para determinar si ha ocurrido un cambio de estado, si RCIF es uno indica que el registro de recepción esta lleno, de esta manera el dato recibido es leído. La palabra recibida es revisada, si esta es un cero indicara que se debe decrementar en

uno el registro acumulador, si la palabra es un uno el registro acumulador no se modificara y si es un dos el registro acumulador se incrementara en uno. El contenido del registro acumulador es enviado al puerto B para que un conversor digital-análogo se encargue de reconstruir la señal análoga. Luego el programa retorna a verificar de nuevo el estado de la bandera RCIF, es decir, a esperar que llegue un nuevo dato.

8.15.5 Diagrama de flujo del receptor. A continuación se presenta el diagrama de flujo del programa principal que fue necesario implementar en el desarrollo de la etapa de recepción para el módulo DELTA.

PROGRAMA PRINCIPAL DEL RECEPTOR DELTA



8.15.6 Programa en código fuente del receptor. A continuación se da el programa detallado, que se utilizo en el microcontrolador, con sus respectivos comentarios.

```
;-------------------------------------------------------------------------------------------------------------------
;                                        Demodulador DELTA (Receptor)
;-------------------------------------------------------------------------------------------------------------------
;       DELTARX.ASM

        list    p=16F874        ;Procesador
        include <p16F874.inc>

;       Al programar el micro tener en cuenta los fusibles de configuración
;       OSC = HS, WDT = OFF, PWRTE = OFF, CP = OFF, BODEN = OFF
;       LVP = Disabled, CPD = OFF, WRT = Enabled, DEBUG = Disabled


;------------------------------------------------- Zona de Registros ----------------------------------------------


tmr_opt         equ     01h     ;
status          equ     03h     ;
puertob         equ     06h     ;
intcon          equ     0bh     ;
pir_pie1        equ     0ch     ;
rc_txsta        equ     18h     ;
txreg_brg       equ     19h     ;Registro de Transmisión / Generador de Baudios
rcreg           equ     1ah     ;Registro de recepción
up_down         equ     20h     ;Acumulador ascendente / descendente
dato            equ     21h     ;Dato recibido


;------------------------------------------------- Asignaciones de bit --------------------------------------------


rp0             equ     5       ;Registro STATUS
gie             equ     7       ;Habilita int. general
peie            equ     6       ;Habilita interrupciones periféricas
rcif            equ     5       ;Bandera de int. por recepción
csrc            equ     7       ;Selecciona fuente de reloj
txen            equ     5       ;Habilita transmisión
rx9             equ     6       ;Habilita recepción de dato de 9 bits
sync            equ     4       ;Selección de modo USART
rx9d            equ     0       ;Noveno bit del dato recibido
spen            equ     7       ;Habilita puerto serial
sren            equ     5       ;Habilita recepción simple
cren            equ     4       ;Habilita recepción continua


;------------------------------------------------------ Definiciones ----------------------------------------------


#define         banco0          bcf     status,rp0
#define         banco1          bsf     status,rp0
```

```
;---------------------------------------------- Configuración del PIC ----------------------------------------------

        org     0               ;Inicio del programa principal

inicio  banco1                  ;Va al banco1
        movlw   00h             ;Configuración de puertos
        movwf   puertob         ;

        movlw   d'16'           ;Configuración de la USART a 294.1 Kbaudios
        movwf   txreg_brg       ;Carga del registro generador de baudios
        bcf     rc_txsta,csrc   ;Modo esclavo
        bsf     rc_txsta,sync   ;Configuración USART como modo sincrónico
        banco0                  ;Retorno al banco0

        bsf     rc_txsta,spen   ;Habilitación del puerto serial
        bcf     rc_txsta,rx9    ;8 bits
        bcf     rc_txsta,sren   ;Deshabilita recepción simple
        bsf     rc_txsta,cren   ;Habilita recepción continua
        bsf     intcon,gie      ;
        bsf     intcon,peie     ;
        bcf     pir_pie1,rcif   ;
        clrf    puertob         ;Inicialización
        clrf    up_down         ;

;---------------------------------------------- PROGRAMA PRINCIPAL ----------------------------------------------

recibir btfss   pir_pie1,rcif   ;Polling, revisión del registro de recepción
        goto    recibir         ;

        movf    rcreg,w         ;Lectura del registro de recepción
        movwf   dato            ;

        movf    dato,w          ;
        btfsc   status,2        ;Dato recibido = 00?
        goto    down            ;

        movlw   01h             ;
        subwf   dato,w          ;
        btfsc   status,2        ;Dato recibido = 01?
        goto    igual           ;

up      incf    up_down,f       ;Dato recibido = 02
        movf    up_down,w       ;Se incrementa el acumulador
        movwf   puertob         ;El acumulador se envía al puertob
        goto    recibir         ;

down    decf    up_down,f       ;Dato recibido = 00
        movf    up_down,w       ;Se decrementa el acumulador
        movwf   puertob         ;El acumulador se envía al puertob
        goto    recibir         ;

igual   movf    up_down,w       ;Dato recibido = 01
        movwf   puertob         ;El acumulador permanece constante
        goto    recibir         ;y se envía al puertob

;-----------------------------------------------------------------------------------------------------------------------
        end
```

8.16 HARDWARE DEL MODULO DELTA

8.16.1 Funcionamiento del transmisor. En el diagrama esquemático de la Figura 53 se puede observar que la señal análoga de entrada es generada por un circuito integrado generador de funciones XR2206, del cual se utilizan dos formas de onda, una onda seno y una triangular, estas pueden ser elegidas por medio de un selector (*dip switch*) ubicado entre los pines 13 y 14 del circuito integrado. También se puede realizar ajustes en la frecuencia y en la amplitud de la señal por medio de un par de potenciómetros (*Trimmers*).

La señal generada es enviada al pin 2 (RA0/AN0) del PIC16F874, esta es la entrada análoga del conversor análogo-digital, la cual esta limitada a un máximo de +5V por la acción de un diodo zener.

El microcontrolador se encarga de realizar el muestreo de la señal análoga, luego cada muestra se compara con el contenido de un registro acumulador que sigue los cambios de la señal análoga, el resultado de la comparación se envía a la línea de transmisión a través del pin 26 (RC7/RX/DT) del PIC, que es el utilizado por la USART para transmitir datos de manera sincrónica, esta señal de salida es del tipo *NRZ unipolar.*

El contenido del acumulador también es modificado por el resultado de la comparación.

Para generar la señal PAM se usa un conversor digital-análogo DAC0832 que se encarga de convertir a un voltaje análogo la muestra que captura el conversor análogo-digital interno del PIC y que se obtiene por el puerto B (RB0 a RB7) del mismo.

A la salida del DAC0832 se utiliza un circuito integrado LF353 con dos amplificadores operacionales, uno de los cuales se utiliza como conversor de Corriente-Voltaje, ya que la salida del DAC es en forma de corriente; y el otro como inversor de voltaje, esto debido a que el primer operacional invierte la salida.

Los puntos de prueba que se utilizan en el módulo transmisor DELTA son cuatro y permiten observar las formas de onda asociadas al proceso de modulación, de la siguiente manera:

- CLK: El cual tiene la señal de sincronismo que utiliza la USART del PIC para realizar la transmisión sincrónica.
- DATO: En este punto se observa la señal DELTA presente en la línea entre el módulo transmisor y el receptor.
- GND: Tierra (se envía al terminal común del Osciloscopio).
- DELTA (PAM): Punto donde se observa la señal PAM que corresponde a la señal análoga de entrada (SEÑAL).
- La entrada análoga se puede obtener del pin 2 (RA0) del microcontrolador PIC16F874.

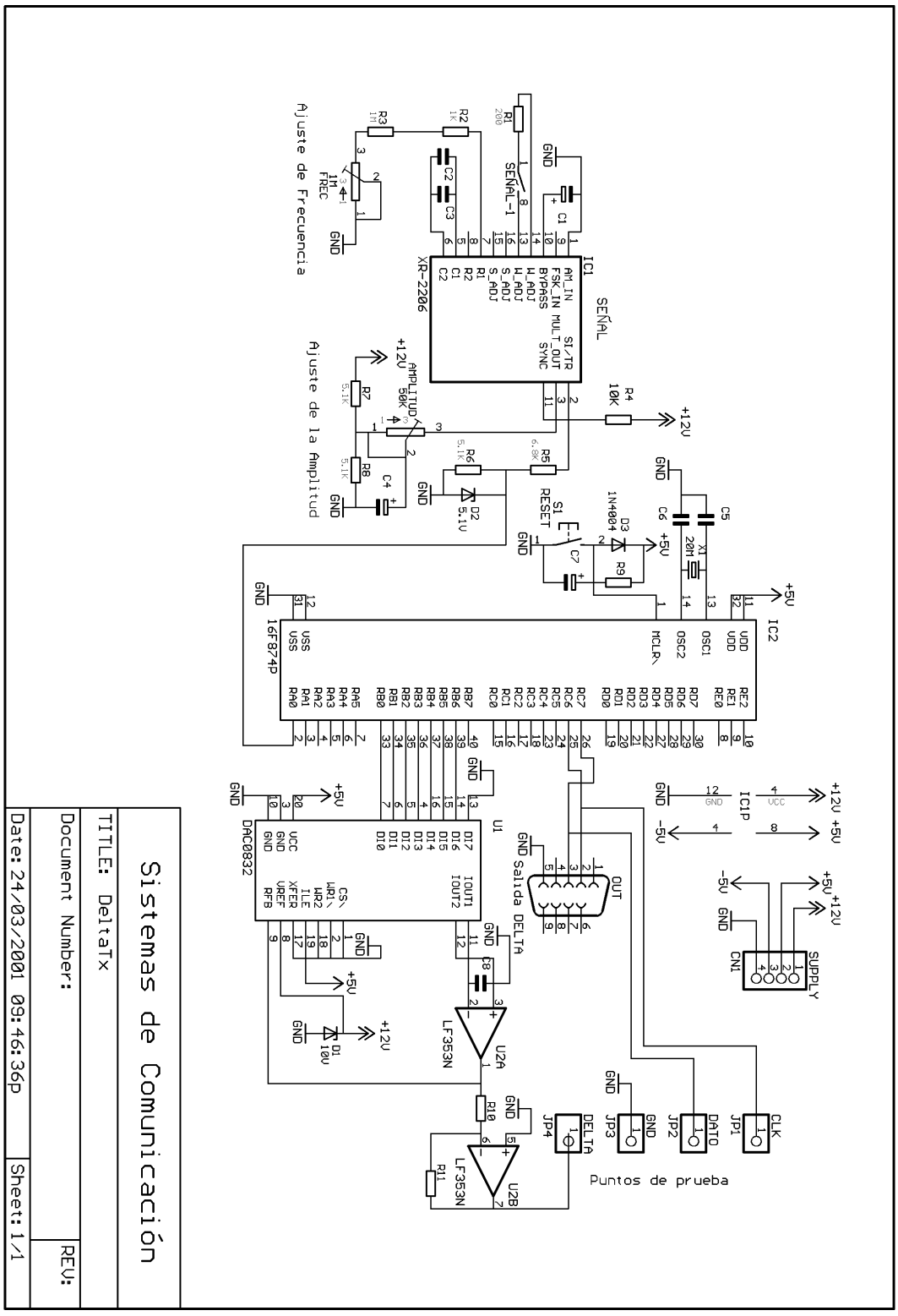
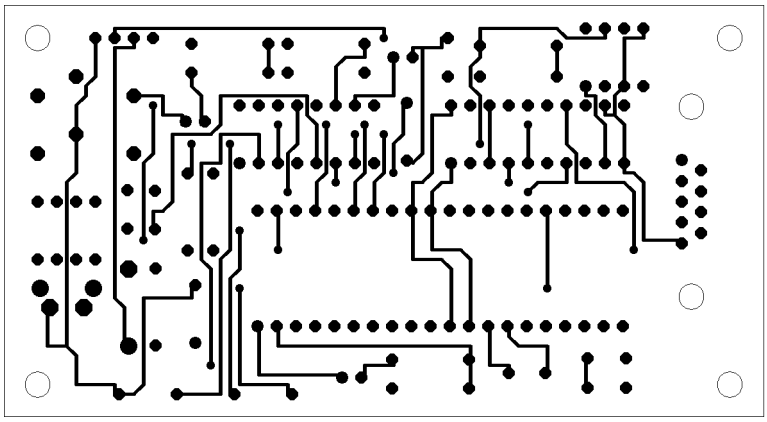
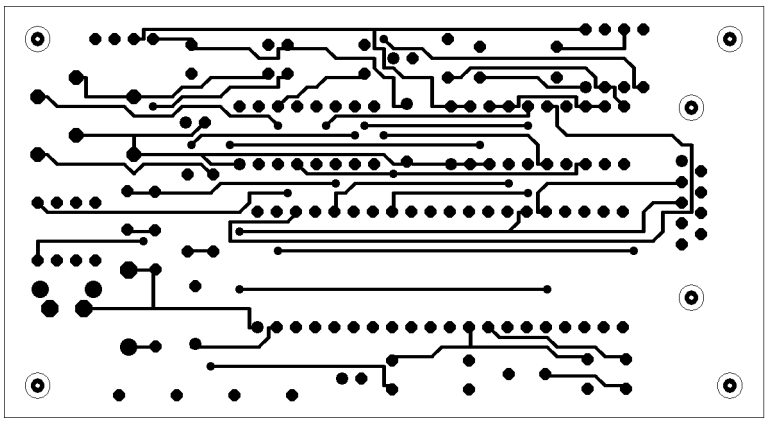


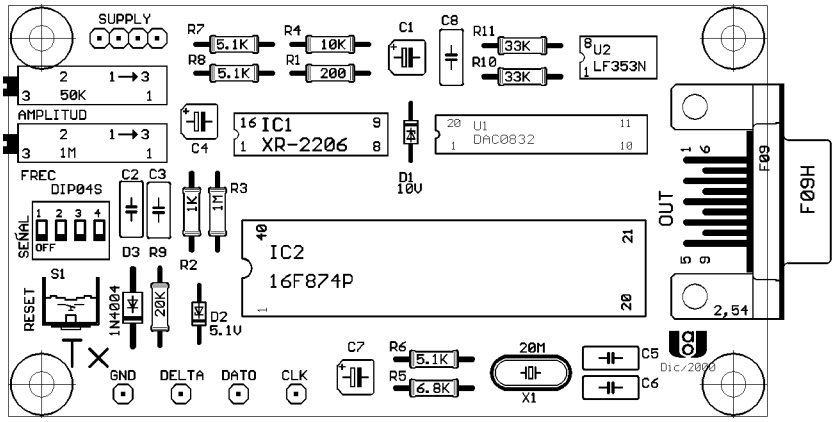
Figura 53 *Diagrama esquemático del transmisor DELTA*



(a) Vista Superior



(b) Vista Inferior



(c) Vista de componentes

Figura 54 *Circuito Impreso del transmisor DELTA*

8.16.2 Funcionamiento del receptor. La señal DELTA de tipo *unipolar NRZ* llega al pin 55 (RC7/RX/DT) del microcontrolador por medio del pin 3 del conector DB9 de entrada.

El PIC16F874 procesa la información serial recibida y de acuerdo a esta se modifica un registro acumulador, cuya función es reproducir los cambios que hace la señal análoga, el contenido del acumulador es enviado a un conversor digital-análogo DAC0832 por medio del puerto B (RB0 a RB7) del PIC. El DAC0832 obtiene entonces una aproximación de la muestra analógica.

A la salida del DAC0832 se utiliza un circuito integrado LF353 con dos amplificadores operacionales, uno de los cuales se utiliza como conversor de Corriente-Voltaje, ya que la salida del DAC es en forma de corriente; y el otro como inversor de voltaje, esto debido a que el primer operacional invierte la salida.

En el módulo receptor DELTA se utiliza cuatro puntos de prueba que permiten observar las formas de onda asociadas a la demodulación, de la siguiente forma:

- CLK: Señal de sincronismo que utiliza la transmisión sincrónica.
- DATO: En este punto se observa la señal DELTA de entrada
- GND: Tierra (se envía al terminal común del Osciloscopio).
- SEÑALOUT: Punto donde se observa la señal análoga recuperada.

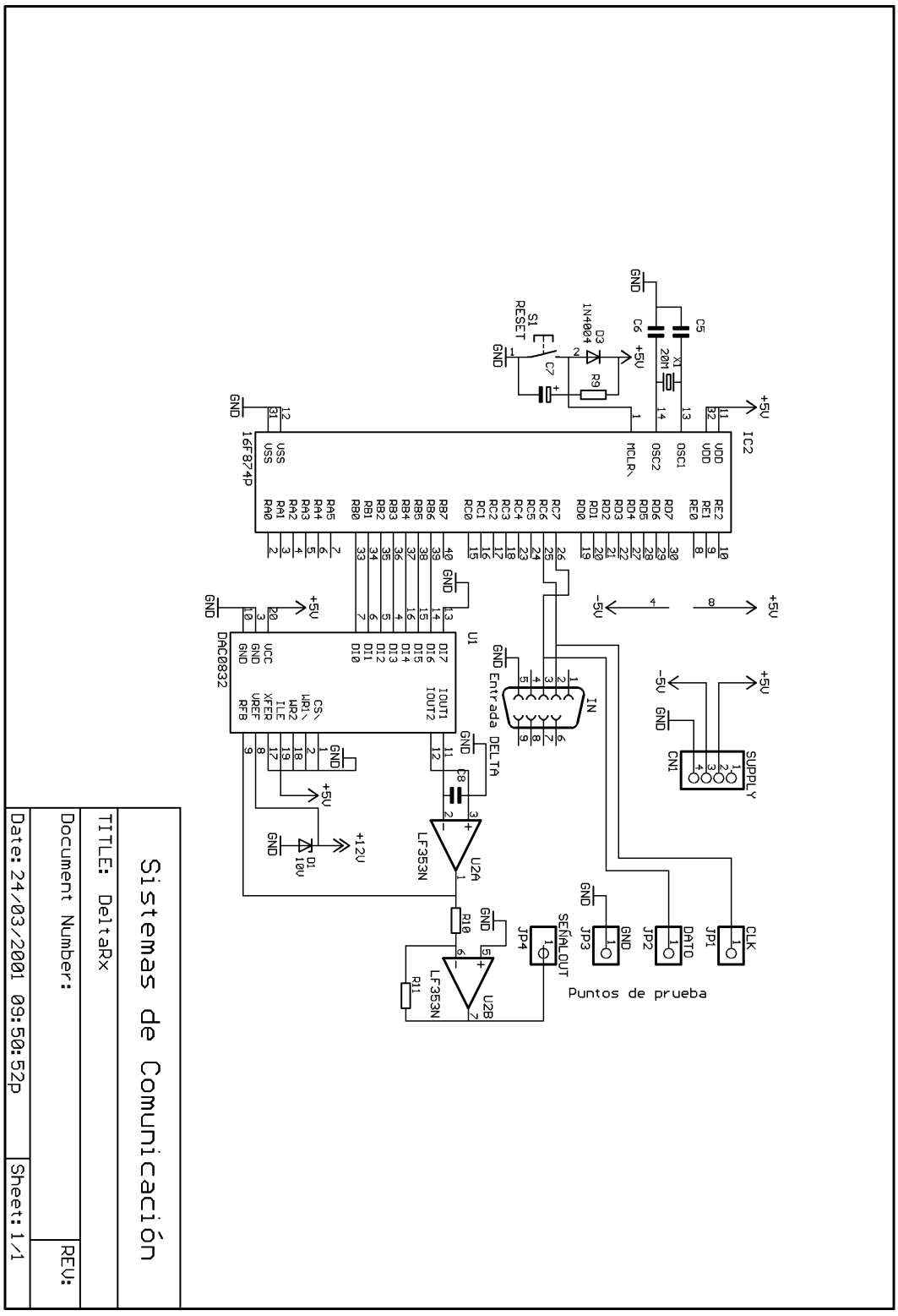
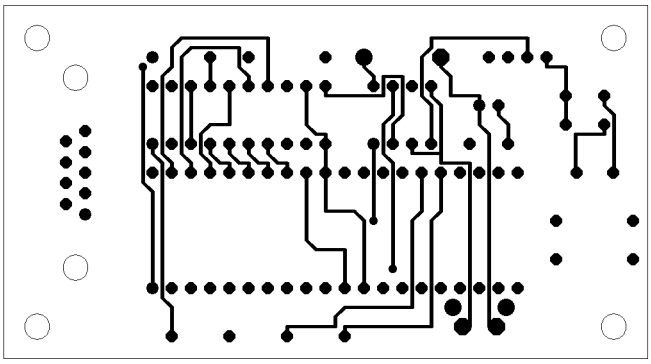
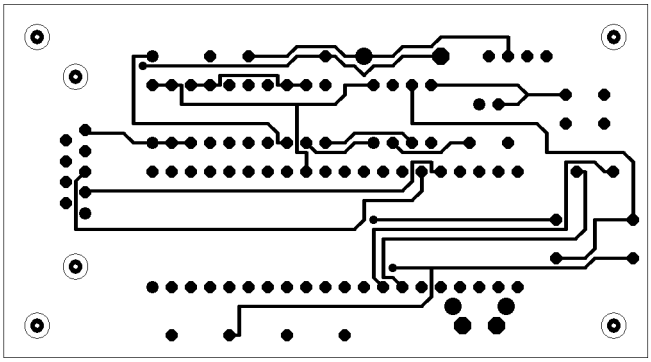


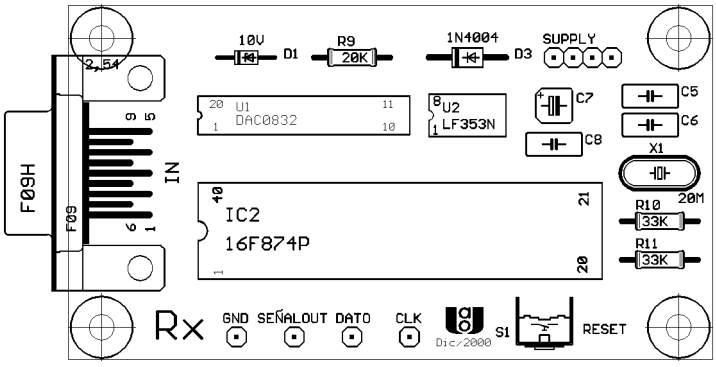
Figura 55 *Diagrama esquemático del receptor DELTA*



(a) Vista Superior



(b) Vista Inferior



(c) Vista de componentes

Figura 56 *Circuito Impreso del receptor DELTA*

8.17 FUENTE DE PODER PARA LOS MODULOS PCM Y DELTA

Los módulos PCM y DELTA necesitan tres niveles de voltaje diferentes.

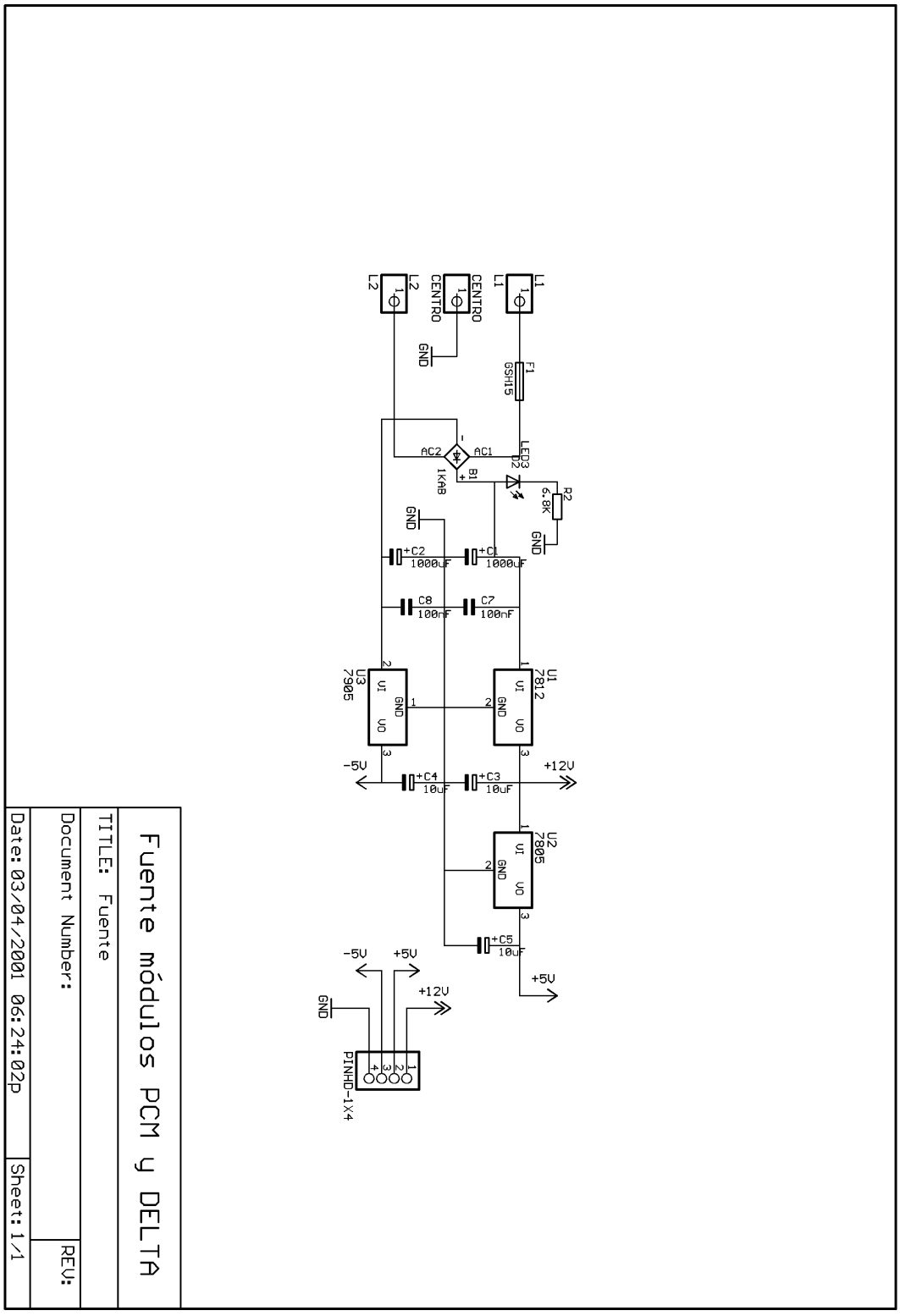
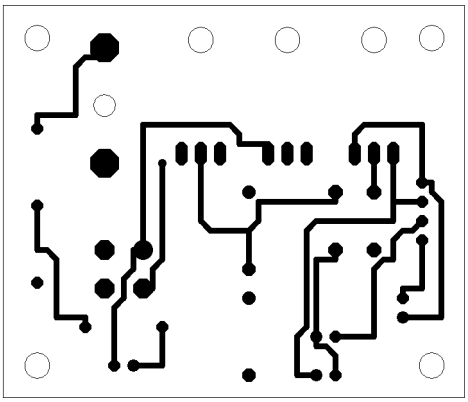
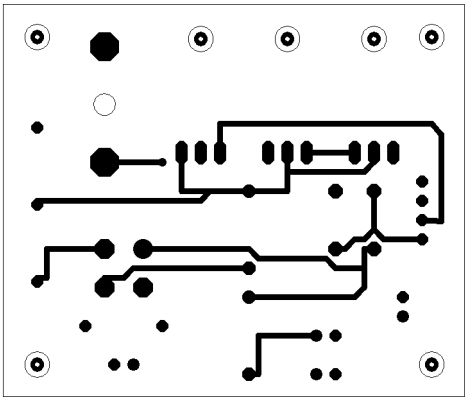


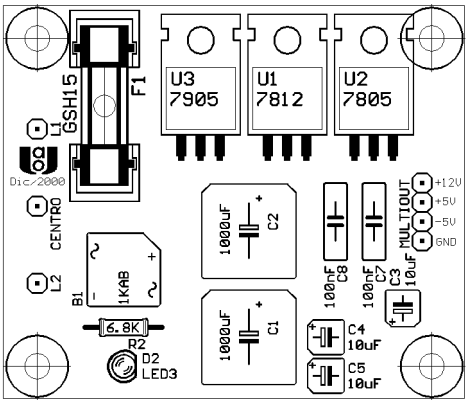
Figura 57 *Diagrama de la fuente de poder utilizada por los módulos PCM y DELTA*



(a) Vista Superior



(b) Vista Inferior



(c) Vista de componentes

Figura 58 *Circuito Impreso de la fuente para los módulos PCM y DELTA*

8.18 LIMITACIONES

Las limitaciones de los módulos con señales digitales de entrada son:

- Transmisión asincrónica con una velocidad de 1200 Baudios.

- Trama de 11 bits (*Start bit,* dato de ocho bits, dos bits de *Stop*).

- Las condiciones de error solo se presentan por tres causas: 1) error en la comunicación entre el Computador y el módulo, 2) error en la comunicación entre los módulos, 3) el Computador inicia una transmisión antes de activar el módulo.

- La forma de onda que generan los moduladores FSK, PSK y ASK no es sinusoidal como se suele observar en los textos de comunicaciones, esto es debido a que se aprovecha todo el ancho de banda del medio de transmisión (cable telefónico), ya que no existe ningún tipo de limitación. La forma sinusoidal de estos sistemas se da por que el ancho de banda del medio de transmisión usualmente se limita para diferentes finalidades, como por ejemplo ajustar el canal al ancho de banda de la voz (en telefonía), este motivo hace que solo se trabaje con la frecuencia fundamental de la señal. Si la señal es un tren de pulsos su frecuencia fundamental tendrá la forma de una onda seno y cabe destacar que el espectro de un tren de pulsos esta conformado por la frecuencia fundamental y sus armónicos.

Las limitaciones de los módulos con señales análogas de entrada son:

- Transmisión sincrónica con una velocidad de 19,53 Kbaudios para el módulo PCM y de 294,1 Kbaudios para el módulo DELTA.

- Trama de nueve bits para el módulo PCM y de ocho bits para el módulo DELTA.

- La máxima frecuencia de la señal análoga de entrada es 100 Hz en el sistema PCM y 30 Hz en el DELTA.

8.19 FORMA DE CONEXION DE LOS MODULOS

8.19.1 Módulos con señal de entrada digital. Para el buen funcionamiento de los módulos con señal de entrada digital (Manchester, FSK, PSK y ASK) se deben conectar de la siguiente forma:

1. Conecte el cable entre el puerto del Computador (COM1 o COM2) que se usara como transmisor y la etapa de transmisión del módulo, el sitio donde se debe conectar el cable en la tarjeta de transmisión esta indicado por el nombre IN.

2. Conecte el cable entre el puerto del Computador (COM1 o COM2) que se usara como receptor y la etapa de recepción del módulo, el sitio donde se debe conectar el cable en la tarjeta de transmisión esta indicado por el nombre OUT.

3. Conecte el cable de comunicación entre la etapa de transmisión y la etapa de recepción del módulo, en la tarjeta de transmisión este sitio esta indicado por el nombre TX y en la tarjeta de recepción esta indicado por el nombre RX.

4. Inserte el cable de alimentación de cada una de las tarjetas que conforman las etapas del módulo en el sitio indicado con el nombre SUPPLY, esta acción activa las tarjetas.

Después de realizar la conexión se puede iniciar la comunicación y la observación de cada una de las señales en las etapas del módulo.

8.19.2 Módulos con señal de entrada análoga. En el caso de los módulos con señal de entrada análoga (PCM y DELTA), para su buen funcionamiento se deben conectar de la siguiente forma:

1. Conecte el cable de comunicación entre la etapa de transmisión y la etapa de recepción del módulo, en la tarjeta de transmisión este sitio esta indicado por el nombre OUT y en la tarjeta de recepción esta indicado por el nombre IN.

2. Inserte el cable de alimentación de cada una de las tarjetas que conforman las etapas del módulo en el sitio indicado con el nombre SUPPLY, esta acción activa las tarjetas.

Después de realizar la conexión se inicia la comunicación y se pueden observar cada una de las señales en las etapas del módulo.

9. APLICACION EN VISUAL BASIC 6.0 (AdCom)

Para administrar las comunicaciones que se dan entre los módulos con entradas digitales se desarrollo una aplicación en lenguaje de programación BASIC utilizando un sistema de desarrollo diseñado para crear aplicaciones graficas como VISUAL BASIC. La aplicación desarrollada permite realizar la transmisión de un carácter a intervalos de tiempo determinados o transmitir un archivo completo con la ayuda del *Hyper-Terminal* de Windows.

Después de iniciar la aplicación *AdCom*, aparecerá la ventana principal, la cual permitirá transmitir datos desde el Computador o visualizar los datos recibidos por el Computador. En la Figura 59 se puede observar el aspecto de la ventana principal de la aplicación *AdCom*. La ventana principal de la aplicación esta dividida en dos secciones RECEPCION y TRANSMISION, cada sección se activa de acuerdo a la función que se va a realizar. El propósito de esta división es el de utilizar la aplicación en los dos Computadores entre los cuales se esta realizando la comunicación para distinguir uno de los dos terminales como el transmisor y el otro como el receptor.

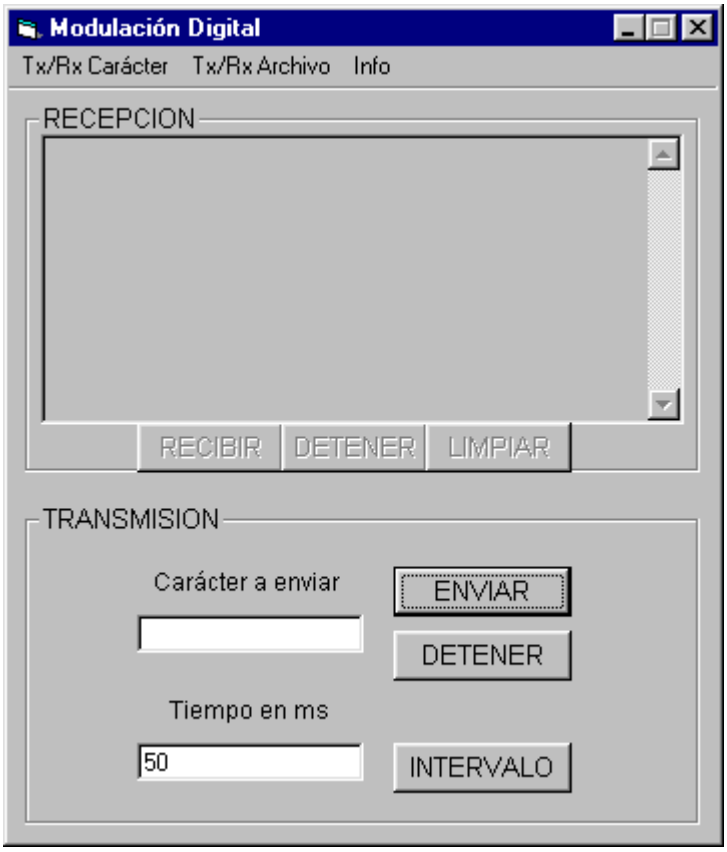


Figura 59 *Ventana principal de la aplicación AdCom (TRANSMISION)*

La función activada por defecto es la de TRANSMISION. Esta sección se encarga de transmitir un solo carácter de manera cíclica a intervalos de tiempo determinados. Dentro de la sección TRANSMISION se encuentran tres botones: ENVIAR, DETENER e INTERVALO, estos botones permiten realizar la transmisión. El botón ENVIAR recoge el ultimo carácter ingresado en la casilla "*Carácter a enviar*" y lo envía por el puerto de comunicaciones, la condición para enviar el carácter es que el puerto este abierto. El botón DETENER detiene la transmisión del carácter, pero no cierra el puerto. El botón INTERVALO recoge el valor contenido en la casilla "*Tiempo en ms*" y lo utiliza como intervalo de tiempo para la transmisión, este tiempo se da en milisegundos y esta limitado a un mínimo de 50ms y un máximo de 64,7 segundos.

Cuando se va a efectuar la función de recepción se activa la sección RECEPCION de la aplicación, como se aprecia en la Figura 60.

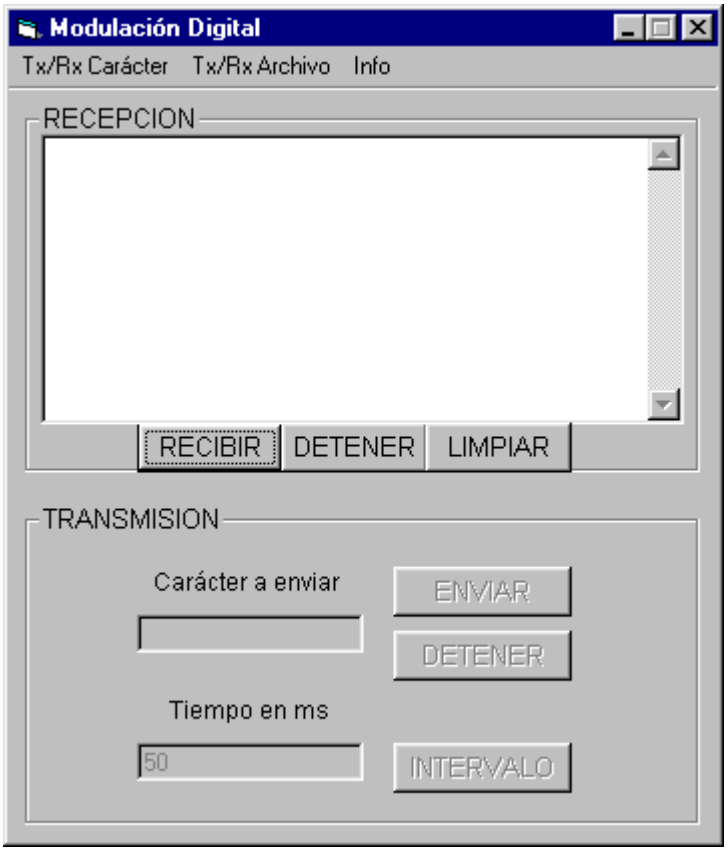


Figura 60 *Sección RECEPCION activada en la aplicación AdCom*

Dentro de la sección RECEPCION se encuentran tres botones, los cuales permiten realizar la recepción de la información que llega por el puerto de comunicaciones. Estos tres botones son: RECIBIR, DETENER y LIMPIAR. El botón RECIBIR toma el carácter que se reciba por el puerto de comunicaciones y lo visualiza en una caja de texto. El botón DETENER detiene la recepción, pero no cierra el puerto. El botón LIMPIAR pone en blanco la caja de texto donde se visualizan los caracteres recibidos.

9.1 MENU TRANSMISION / RECEPCION DE UN CARACTER

Este menú se encarga de activar las secciones de TRANSMISION y RECEPCION de la ventana principal estas permiten transmitir o recibir caracteres entre dos Computadores, utilizando el puerto de comunicaciones. El menú "*Tx/Rx Carácter*" contiene las siguientes opciones (ver Figura 61):

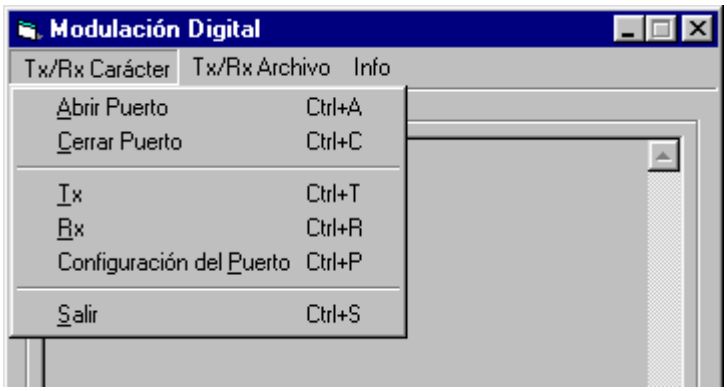


Figura 61 *Opciones del menú "Tx/Rx Carácter"*

- **Abrir Puerto**

Esta opción permite abrir un puerto de comunicaciones para establecer una comunicación con otro Computador. El estado del puerto se aprecia en la barra de titulo de la aplicación (ver Figura 62).

Figura 62 *Mensaje de puerto abierto en la barra de titulo de la aplicación*

- **Cerrar Puerto**

Esta opción permite cerrar un puerto de comunicaciones para eliminar una comunicación establecida con otro Computador. El estado del puerto se aprecia en la barra de titulo de la aplicación (ver Figura 63).

Figura 63 *Mensaje de puerto cerrado en la barra de titulo de la aplicación*

- **Tx**

La opción *Tx* activa la sección de la ventana principal que permite realizar la transmisión cíclica de un carácter.

- **Rx**

La opción *Rx* activa la sección de la ventana principal que permite realizar la recepción de los caracteres que lleguen al puerto de comunicaciones para visualizarlos en una caja de texto.

- **Configuración del puerto**

Esta opción despliega la ventana de la Figura 64, en la cual se pueden observar las características del puerto de comunicaciones y se puede elegir el puerto de comunicación a utilizar (COM1 o COM2). Si no se elige puerto de comunicaciones se utiliza COM1 como puerto por defecto. Después de ejecutar la opción "*Abrir puerto*" la opción de "*Configuración del puerto*" desaparece, para que esta vuelva aparecer se debe ejecutar "*Cerrar puerto*".

Figura 64 *Ventana de Configuración del puerto de comunicación*

- **Salir**

Esta opción termina la ejecución de la aplicación.

9.2 MENU TRANSMISION / RECEPCION DE UN ARCHIVO

Este menú permite realizar la transmisión de un archivo entre dos Computadores, esta transmisión se realiza con la ayuda de la aplicación *Hyper-Terminal* de Windows. El menú "*Tx/Rx Archivo*" esta compuesto por las siguientes opciones (ver Figura 65):

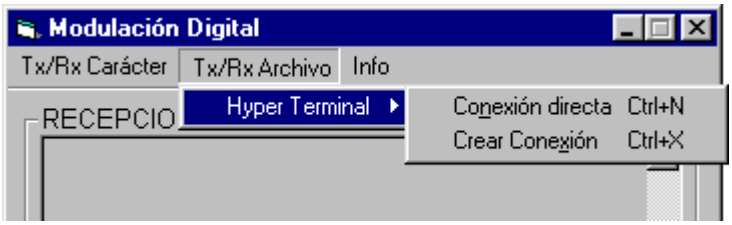


Figura 65 *Opciones del menú "Tx/Rx Archivo"*

- **Conexión directa**

La opción "*Conexión directa*" crea una nueva conexión en el *Hyper-Terminal* con el nombre de "*Transferencia de archivos*" (ver Figura 66), la configuración del puerto para esta conexión esta predeterminada de la siguiente manera: se usa por defecto el puerto COM1, la velocidad de transmisión es de 1200 bits por segundo, se emplean ocho bits de datos, no se utiliza bit de paridad y tampoco control de flujo. Si se desea utilizar el puerto COM2 se debe acudir a la opción Propiedades (Archivo > Propiedades) del *Hyper-Terminal* y seleccionarlo.

Después de dar clic en la opción "*Conexión directa*" se debe esperar unos segundos a que la conexión sea creada, para establecer una comunicación (conectar) se debe abrir el puerto ejecutando Llamar > Llamar.

Figura 66 *Ventana "Transferencia de archivos" creada en el Hyper-Terminal*

- **Crear conexión**

Esta opción ejecuta la aplicación *Hyper-Terminal* y permite al usuario crear su propia conexión (ver Figura 67). Es importante anotar que para usar los módulos se debe tener en cuenta sus características de comunicación.

Figura 67 *Ventana principal del Hyper-Terminal*

9.3 MENU INFO

Este menú despliega una ventana con información general de la aplicación *AdCom* (ver Figura 68).

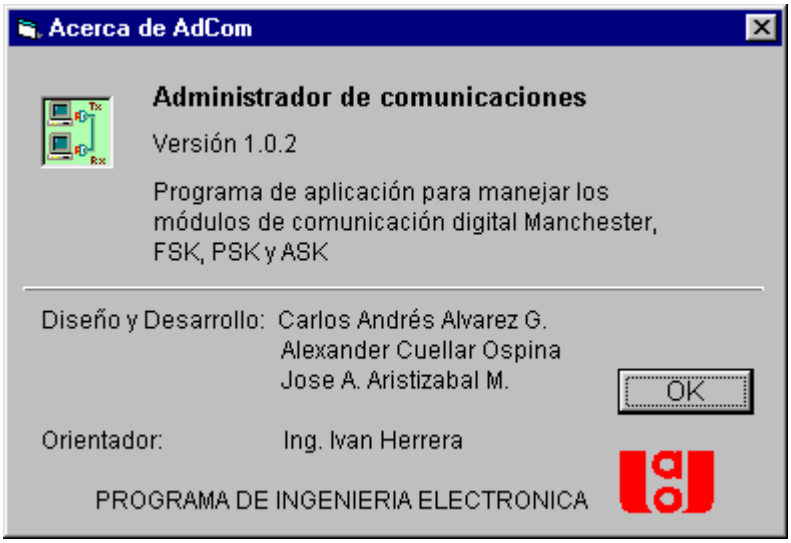


Figura 68 *Ventana de información general sobre la aplicación AdCom*

9.4 FORMA DE REALIZAR UNA TRANSMISION O UNA RECEPCION

9.4.1 Transmisión o recepción de un carácter. Los pasos a seguir para realizar la transmisión cíclica de un carácter son:

1. Activar la sección de TRANSMISION de la aplicación (*Tx/Rx Carácter > Tx)*.

2. Establecer el puerto de comunicaciones y el tiempo del intervalo de la transmisión, si estos no son determinados se usa por defecto el puerto COM1 y un tiempo de 50ms.

3. Abrir el puerto para establecer la comunicación.

4. Ingresar el carácter a transmitir y hacer clic en el botón ENVIAR para iniciar la transmisión.
5. Si se desea modificar el carácter enviado, se debe ingresar uno nuevo en la casilla "*Carácter a* enviar" y hacer clic en el botón ENVIAR.

6. Si se desea modificar el tiempo del intervalo de transmisión, se debe ingresar el nuevo valor en la casilla "*Tiempo en ms*" y hacer clic en el botón INTERVALO.

7. Para detener la transmisión se debe hacer clic en el botón DETENER.

Los pasos a seguir para realizar la recepción de caracteres son:

1. Activar la sección de RECEPCION de la aplicación (*Tx/Rx Carácter > Rx)*.

2. Establecer el puerto de comunicaciones, si este no es determinado se usa por defecto el puerto COM1.

3. Abrir el puerto de comunicaciones para establecer la comunicación.

4. Hacer clic en el botón RECIBIR para iniciar la recepción, los caracteres recibidos son visualizados en la caja de texto.

5. Si desea borrar el contenido de la caja de texto se debe hacer clic en el botón LIMPIAR.

6. Para detener la recepción se debe hacer clic en el botón DETENER.

9.4.2 Transmisión o recepción de un archivo. Para realizar una transmisión o una recepción de un archivo se utiliza el *Hyper-Terminal*, este puede ser ejecutado desde la aplicación *Adcom* o desde Inicio > Programas > Accesorios > Comunicaciones > Hyper terminal.

En el caso en que se use la aplicación *AdCom* para ejecutar el *Hyper-Terminal* se pueden utilizar dos opciones: "*Conexión directa*" que crea una conexión con las características de comunicación necesarias para los módulos, o "*Crear*

conexión” que abre el *Hyper-Terminal* para que el usuario cree su propia conexión.

Los pasos a seguir para realizar la transmisión de un archivo son:

1. Activar el *Hyper-Terminal* y crear una nueva conexión con las características de comunicación de los módulos. Se puede usar la opción “*Conexión directa*” (Tx/Rx Archivo > Hyper-Terminal > Conexión directa) de la aplicación *AdCom*.

2. Abrir el puerto (conectar) ejecutando la opción *Llamar* (Llamar > Llamar) para establecer la comunicación, esto en el caso de que no este abierto.
3. Para seleccionar el archivo que se va a enviar se debe elegir la opción “*Enviar archivo...*” (Transferir > Enviar archivo...) del *Hyper-Terminal*, esta opción despliega una ventana en la cual se da la ubicación del archivo a enviar y también se elige el protocolo de comunicación que se va a utilizar. Por ultimo se hace clic en el botón *Enviar* para iniciar la transmisión, esto con la condición de que el receptor también este configurado y a la espera de una transmisión.

Los pasos a seguir para realizar la recepción de un archivo son:

1. Activar el *Hyper-Terminal* y crear una nueva conexión con las características de comunicación de los módulos. Se puede usar la opción “*Conexión directa*” (Tx/Rx Archivo > Hyper-Terminal > Conexión directa) de la aplicación *AdCom*.

2. Abrir el puerto (conectar) ejecutando la opción *Llamar* (Llamar > Llamar) para establecer la comunicación, esto en el caso de que no este abierto.

3. Para recibir un archivo se debe elegir la opción "*Recibir archivo...*" (Transferir > Recibir archivo...) del *Hyper-Terminal*, esta opción despliega una ventana en la cual se da la ubicación en donde va ser alojado el archivo recibido, también se debe elegir el protocolo de comunicación a utilizar, este ultimo debe coincidir con el protocolo utilizado por el transmisor. Por ultimo se hace clic en el botón *Recibir* para iniciar la recepción.

10. CONCLUSIONES

1. Al termino del proyecto se dejó una base practica para que el estudiante refuerce los conocimientos teóricos adquiridos en la asignatura “Sistemas de Comunicaciones 1”.

2. El proyecto se convierte en un aporte para el Programa de Ingeniería Electrónica de la Corporación Universitaria Autónoma de Occidente con miras a fortalecer sus laboratorios.

3. Se brinda a los estudiantes la oportunidad de observar, analizar y sacar conclusiones de cada uno de los sistemas de modulación utilizados.

4. Se resalta la importancia de los microcontroladores en diferentes aplicaciones para las comunicaciones

5. Los módulos realizados requirieron bastante tiempo para su desarrollo e implementación lo que no permitió culminar los módulos “Analizador de espectro” y “Modem FSK” cuya complejidad era mayor.

BIBLIOGRAFIA

ANGULO, José M[a] y ANGULO, Ignacio. *Microcontroladores PIC "Diseño practico de aplicaciones".* Madrid: Mc Graw-Hill / Interamericana de España, 1997. 221 p.

COUCH II, León W. *Sistemas de comunicación digitales y analógicos.* 5[a] edición. México: Prentice-Hall, Inc., 1998. 742 p.

PEEBLES, Peyton Z. *Digital communication systems.* New Jersey: Prentice-Hall, Inc., 1987. 432 p.

STALLINGS, William. *Comunicaciones y redes de computadores.* 5[a] edición, Madrid: Prentice-Hall Iberia, 1997. 792 p.

TOMASI, Wayne. *Advanced electronic communication systems.* New Jersey: Prentice-Hall, Inc., 1987. 373 p.

-------- . *Sistemas de comunicaciones.* 2[a] edición, México: Prentice-Hall, Inc., 1996. 858 p.

ANEXOS

Anexo A

GUIA DE USO DE LOS MODULOS MANCHESTER, FSK, PSK Y ASK

A continuación se describen cada una de las partes de las etapas de transmisión y recepción que utilizan los sistemas Manchester, FSK, PSK y ASK, así como la forma de conexión y la manera de ingresar la señal de entrada digital.

1. LOCALIZACION DE PARTES EN LA ETAPA DE TRANSMISION

La tarjeta que recibe la señal del Computador y realiza el proceso de codificación o modulación esta compuesta como se aprecia en la Figura 1.

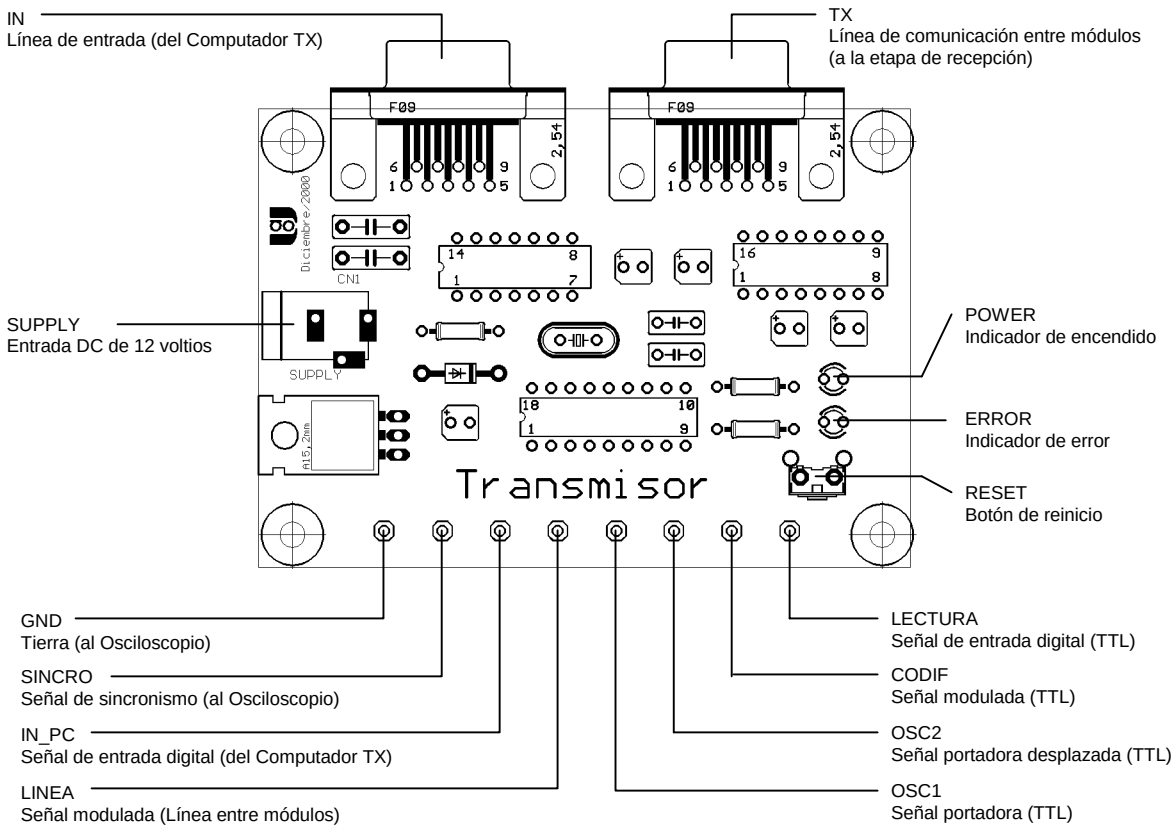


FIGURA 1 *Etapa de transmisión de los módulos Manchester, FSK, PSK y ASK*

1.1 Conectores

SUPPLY:
Esta es la entrada a la cual se debe conectar un adaptador de 12 voltios DC, el cual se encargara de alimentar la tarjeta.

IN:
En este punto se debe conectar la línea que proviene del puerto (COM1 o COM2) del Computador transmisor.

TX:
Es la línea de comunicación entre las etapas de transmisión y recepción de los módulos, esta línea va a la etapa de recepción.

1.2 Indicadores

POWER:
Este es un diodo LED de color amarillo, el cual se enciende en el momento de activar la etapa de transmisión para indicar la presencia del voltaje de alimentación.

ERROR:
Este indicador es un diodo LED de color rojo, el cual se encarga de indicar que se presento un error de comunicación entre el Computador y la etapa de transmisión.

1.3 Controles

RESET:
Este control es un micropulsador, el cual permite reiniciar el programa que esta corriendo en el microcontrolador de la etapa de transmisión, esto en caso que sea necesario hacerlo.

1.4 Puntos de prueba

GND:

Tierra, terminal común de la etapa de transmisión, este se debe conectar a la tierra del Osciloscopio.

SINCRO:

Señal de sincronización por flanco de subida para el Osciloscopio, la sincronización ocurre cada vez que se presenta una trama de datos. Este terminal se debe enviar a la entrada de sincronización externa del Osciloscopio.

IN_PC:

Señal digital de entrada (trama de datos) que proviene del computador que esta realizando la transmisión.

LINEA:

En este terminal se puede observar la trama de datos modulada, o codificada en el caso del módulo Manchester, que se envía a la etapa de recepción a través de la línea de comunicación.

OSC1:

La señal que se obtiene de este punto es la que pertenece a la modulación o codificación de un uno lógico de la trama de datos (señal portadora), esta señal tiene niveles de voltaje TTL.

OSC2:

En este punto se obtiene la señal que pertenece a la modulación o codificación de un cero lógico de la trama de datos (señal portadora desplazada), esta señal tiene niveles de voltaje TTL.

CODIF:

En este punto se puede observar la señal modulada o codificada de la trama de datos en niveles de voltaje TTL.

LECTURA:

Este punto permite observar la entrada digital (trama de datos) que proviene del Computador transmisor.

2. LOCALIZACION DE PARTES EN LA ETAPA DE RECEPCION

La tarjeta que recibe la señal de la etapa de transmisión y realiza el proceso de decodificación o demodulación esta compuesta como se aprecia en la Figura 2.

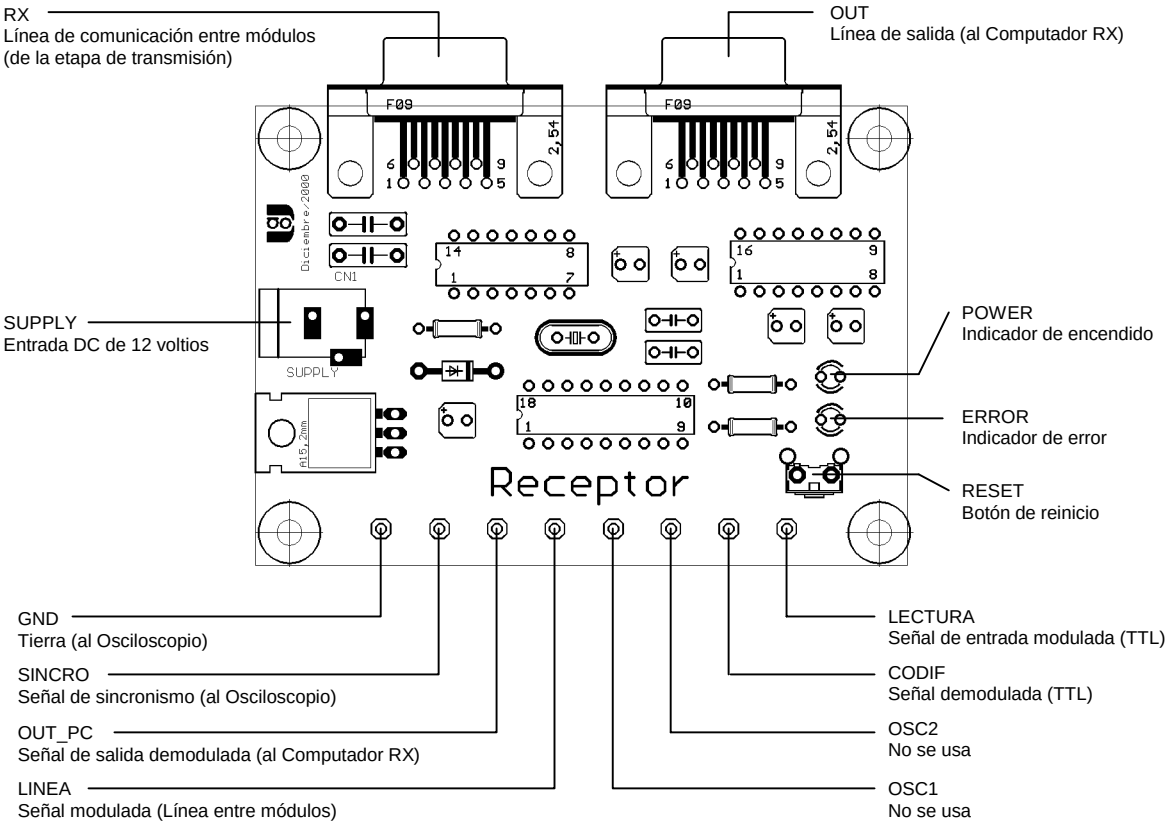


FIGURA 2 *Etapa de recepción de los módulos Manchester, FSK, PSK y ASK*

2.1 Conectores

SUPPLY:

Esta es la entrada a la cual se debe conectar un adaptador de 12 voltios DC, el cual se encargara de alimentar la tarjeta.

RX:

Es la línea de comunicación entre las etapas de transmisión y recepción de los módulos, esta línea viene de la etapa de transmisión.

OUT:

En este punto se debe conectar la línea que se envía al puerto (COM1 o COM2) del Computador receptor.

2.2 Indicadores

POWER:

Este es un diodo LED de color amarillo, el cual se enciende en el momento de activar la etapa de recepción para indicar la presencia del voltaje de alimentación.

ERROR:

Este indicador es un diodo LED de color rojo, el cual se encarga de indicar que se presento un error de comunicación entre la etapa de transmisión y la etapa de recepción.

2.3 Controles

RESET:

Este control es un micropulsador, el cual permite reiniciar el programa que esta corriendo en el microcontrolador de la etapa de recepción, esto en caso que sea necesario hacerlo.

2.4 Puntos de prueba

GND:

Tierra, terminal común de la etapa de recepción, este se debe conectar a la tierra del Osciloscopio.

SINCRO:

Señal de sincronización por flanco de subida para el Osciloscopio, la sincronización ocurre cada vez que se presenta una trama de datos modulada o codificada. Este terminal se debe enviar a la entrada de sincronización externa del Osciloscopio.

OUT_PC:

Señal demodulada de salida (trama de datos recuperada) que se envía al puerto del Computador que esta realizando la recepción.

LINEA:

En este terminal se puede observar la trama de datos modulada, o codificada en el caso del módulo Manchester, que llega a la etapa de recepción a través de la línea de comunicación.

OSC1:

Este punto no se usa.

OSC2:

Este punto no se usa.

DECODIF:

En este punto se puede observar la señal demodulada o decodificada (recuperación de la trama de datos) en niveles de voltaje TTL.

LECTURA:

Este punto permite observar la señal modulada que proviene de la etapa de transmisión en niveles de voltaje TTL.

3. FUNCIONAMIENTO DE LOS MODULOS

3.1 Conexión

1. Conecte el adaptador de alimentación al toma DC de 12 voltios de cada una de las etapas. El indicador de alimentación (POWER) se encenderá.

2. Conecte la línea que proviene del puerto del Computador que se utiliza como transmisor al conector de la etapa de transmisión indicado con el nombre IN.

3. Conecte la línea de comunicación entre la etapa de transmisión y la etapa de recepción. Esta línea se conecta entre el sitio señalado con el nombre TX en la etapa de transmisión y el indicado con el nombre RX en la etapa de recepción.

4. Conecte la línea que proviene del conector indicado con el nombre OUT de la etapa de recepción al puerto del Computador que realiza la recepción.

5. Conecte el terminal señalado con el nombre GND al terminal de tierra del Osciloscopio, de esta manera se podrán observar las señales generadas por las etapas de los módulos.

6. Conecte el terminal SINCRO a la entrada de sincronización externa del Osciloscopio y configure el Osciloscopio para que la sincronización se realice por un flanco de subida.

7. Envié la punta de prueba del Osciloscopio a uno de los puntos donde se obtienen las señales a observar en las etapas.

Después de conectar y activar las etapas de transmisión y recepción de los módulos, el indicador de alimentación (POWER) se encenderá y el indicador de error de comunicación (ERROR) deberá apagarse en ambas etapas. Si el indicador ERROR permanece encendido en cualquiera de las etapas, se reiniciara el programa que esta corriendo en el microcontrolador con la ayuda del botón de reinicio (RESET).

3.2 Entrada digital

La entrada digital es una trama de datos que proviene de uno de los puertos (COM1 o COM2) del Computador que realiza la transmisión y que utiliza el protocolo RS-232. Para acceder a estos puertos se debe utilizar una aplicación o programa que permita manipularlos. Esta aplicación puede ser el *Hyper-Terminal* que viene incluido en Windows o la aplicación *AdCom 1.0.2* creada especialmente para manipular la trama de datos que se utiliza como entrada digital para los módulos. La trama de datos se transmite a una velocidad de 1200 Baudios y esta compuesta por ocho bits de datos, dos bits de *Stop* y sin bit de paridad (1200,8,N,2).

La aplicación *AdCom* permite transmitir de manera cíclica un carácter a través de uno de los puertos del Computador. El *Hyper-Terminal* permite transmitir caracteres y archivos.

3.2.1 Utilización del AdCom

Esta aplicación se utiliza para transmitir un carácter de manera cíclica a otro Computador. Para utilizar esta aplicación se deben seguir los siguientes pasos:

1. Conecte la etapa de transmisión al puerto del Computador que va a realizar la transmisión.

2. Conecte la etapa de recepción al puerto del Computador que se usara como receptor.

3. Inicie la aplicación *AdCom* en los Computadores utilizados (Inicio > Programas > AdCom).

4. En el Computador que se usa como transmisor active la sección TRANSMISION de la aplicación (Tx/Rx Carácter > Tx). Establezca el puerto de comunicaciones y el tiempo del intervalo de transmisión, si estos no son determinados se usa por defecto el puerto COM1 y un tiempo de 50ms.

5. En el Computador que se usa como receptor active la sección RECEPCION de la aplicación (Tx/Rx Carácter > Rx). Establezca el puerto de comunicaciones, si este no es determinado se usa por defecto el puerto COM1.

6. Abra el puerto de comunicaciones en cada uno de los Computadores utilizados (Tx/Rx Carácter > Abrir puerto) para establecer la comunicación.

7. En el Computador transmisor ingrese el carácter que se va a transmitir en la casilla "*Carácter a enviar*" y presione el botón ENVIAR para iniciar la transmisión.

8. En el Computador receptor presione el botón RECIBIR para iniciar la recepción de los caracteres que llegan al puerto, cada carácter recibido se visualiza en la pantalla.

9. Para detener la transmisión o la recepción presione el botón DETENER en el Computador en el cual vaya a realizar la acción.

Si se desea modificar el carácter que el Computador transmisor esta enviando ingrese uno nuevo en la casilla "*Carácter a enviar*" y presione el botón ENVIAR. Para modificar el intervalo de transmisión del carácter ingrese uno nuevo en la casilla "*Tiempo en ms*" y presione el botón intervalo.

3.2.2 Utilización del Hyper Terminal

El *Hyper-Terminal* se usa para transmitir archivos completos a otro Computador. Para utilizar esta aplicación se deben seguir los siguientes pasos:

1. Conecte la etapa de transmisión al puerto del Computador que va a realizar la transmisión.

2. Conecte la etapa de recepción al puerto del Computador que se usara como receptor.

3. Inicie la aplicación *Hyper-Terminal* en los Computadores utilizados (Inicio > Programas > Accesorios > Comunicaciones > Hyper Terminal) y cree una nueva conexión con las características de comunicación de las etapas. También se puede usar la aplicación *AdCom* para crear automáticamente la conexión (Tx/Rx Archivo > Hyper Terminal > Conexión directa).

4. Abra el puerto (conectar) en cada uno de los Computadores ejecutando la opción *Llamar* (Llamar > Llamar) de esta manera se establece la comunicación.

5. En el Computador receptor elija la opción *Recibir archivo*, esta opción despliega una ventana en la cual se da la ubicación en donde va a ser alojado el archivo recibido y el protocolo de comunicación que se va a utilizar. Para iniciar la recepción presione el botón RECIBIR.

6. Seleccione el archivo que se va a enviar desde el Computador transmisor ejecutando la opción *Enviar Archivo* (Transferir > Enviar archivo...), esta opción despliega una ventana en la cual se da la ubicación del archivo y el protocolo de comunicación que se va a utilizar. Para iniciar la transmisión presione el botón ENVIAR.

Cuando se elige la opción *Conexión directa* (Tx/Rx Archivo > Hyper Terminal > Conexión directa) de la aplicación *AdCom* el puerto de comunicación que se utiliza por defecto es el COM1, si se quiere utilizar otro puerto se debe acudir a la opción *Propiedades* (Archivo > Propiedades) y seleccionarlo.

También es posible transmitir caracteres con el *Hyper-Terminal*, para esto se ingresan los caracteres en la ventana del *Hyper-Terminal* activo en el Computador transmisor, a medida que los caracteres son ingresados estos son transmitidos. El Computador receptor toma los caracteres que llegan al puerto y los visualiza en pantalla a través de la ventana del *Hyper-Terminal*.

GUIA DE USO DEL MODULO PCM

A continuación se describen cada una de las partes de las etapas de transmisión y recepción que utiliza el sistema PCM, así como su forma de conexión.

1. LOCALIZACION DE PARTES EN LA ETAPA DE TRANSMISION

La tarjeta que genera la señal análoga de entrada y realiza el proceso de modulación de la misma esta compuesta como se aprecia en la Figura 1.

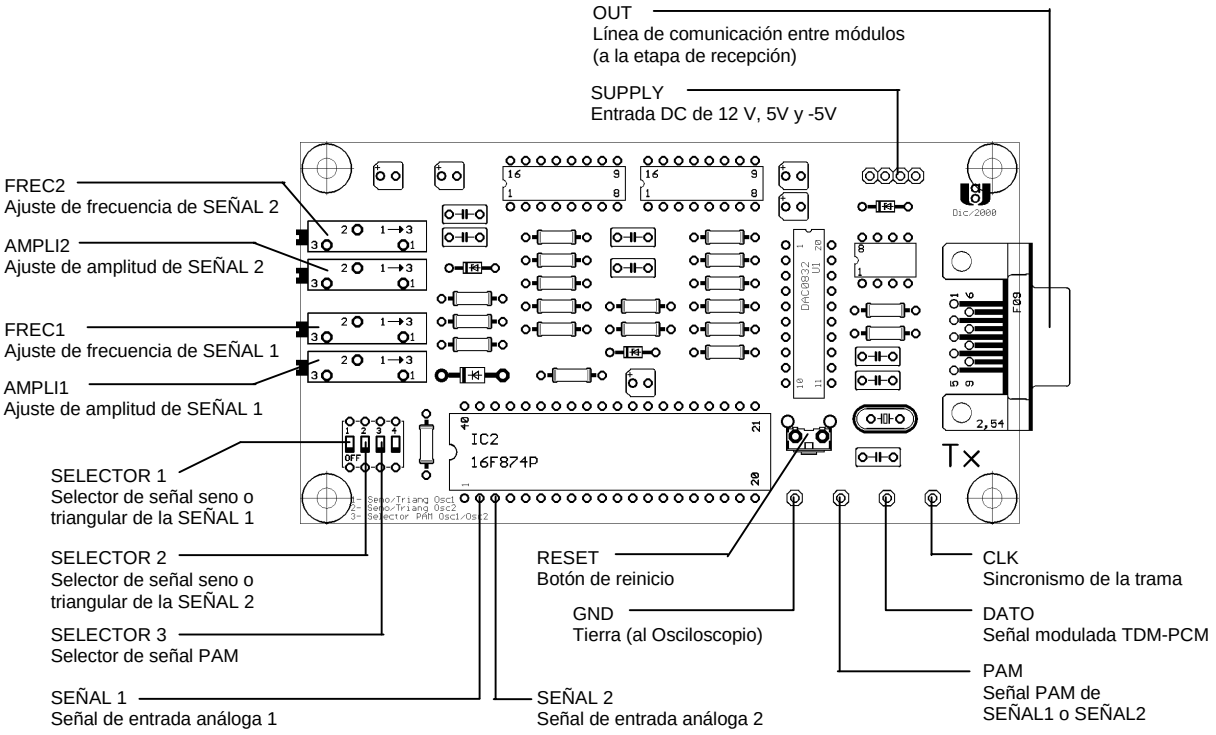


FIGURA 1 *Etapa de transmisión del módulo PCM*

1.1 Conectores

SUPPLY:

Esta es la entrada a la cual se debe conectar la fuente de poder que genera tres voltajes diferentes (12, 5 y -5 voltios), estos se encargan de alimentar la tarjeta.

OUT:

Es la línea de comunicación entre las etapas de transmisión y recepción del módulo, esta línea va a la etapa de recepción.

1.2 Controles

RESET:

Este control es un micropulsador, el cual permite reiniciar el programa que esta corriendo en el microcontrolador de la etapa de transmisión, esto en caso que sea necesario hacerlo.

AMPLI1:

Este control es un potenciómetro (Trimmer), el cual permite ajustar la amplitud de la SEÑAL 1. La máxima amplitud es de cinco voltios.

FREC1:

Este control es un potenciómetro (Trimmer) que permite realizar el ajuste de la frecuencia de la SEÑAL 1, esta frecuencia se puede variar entre el rango 50 y 100 Hz.

AMPLI2:

Este control es un potenciómetro (Trimmer), el cual permite ajustar la amplitud de la SEÑAL 2. La máxima amplitud es de cinco voltios.

FREC2:

Este control es un potenciómetro (Trimmer) que permite realizar el ajuste de la frecuencia de la SEÑAL 2, esta frecuencia se puede variar entre el rango de 50 y 100 Hz.

1.3 Selectores

SELECTOR 1:

Este selector es un switch de dos posiciones, el cual permite elegir la forma de onda de la SEÑAL 1, es decir, una onda seno o una triangular.

SELECTOR 2:

Este selector permite elegir la forma de onda de la SEÑAL 2, es decir, se elige si la SEÑAL 2 será una onda seno o una triangular.

SELECTOR 3:

Con este selector se elige la señal análoga que genera la señal PAM, esta señal se obtiene del terminal señalado con el nombre PAM.

1.4 Puntos de prueba

GND:

Tierra, terminal común de la etapa de transmisión, este se debe conectar a la tierra del Osciloscopio.

PAM:

En este punto se obtiene la señal PAM correspondiente a una de las dos señales análogas (SEÑAL 1 y SEÑAL 2) que se usan como entrada de la etapa de transmisión.

DATO:

Este punto permite observar la señal modulada, es decir, la señal PCM que corresponde a la muestra de la señal análoga, en este caso se observaran dos señales PCM multiplexadas (trama TDM-PCM) ya que son dos señales análogas. Esta señal es la que se envía a la etapa de recepción a través de la línea de comunicación.

CLK:

En este terminal se puede observar la señal de sincronización que acompaña la trama TDM-PCM, ya que se utiliza comunicación sincrónica.

Las señales análogas, *SEÑAL 1* y *SEÑAL 2*, se obtienen en los pines dos y tres respectivamente, del microcontrolador PIC16F874.

2. LOCALIZACION DE PARTES EN LA ETAPA DE RECEPCION

La tarjeta que recibe la señal de la etapa de transmisión y realiza el proceso de demodulación esta compuesta como se aprecia en la Figura 2.

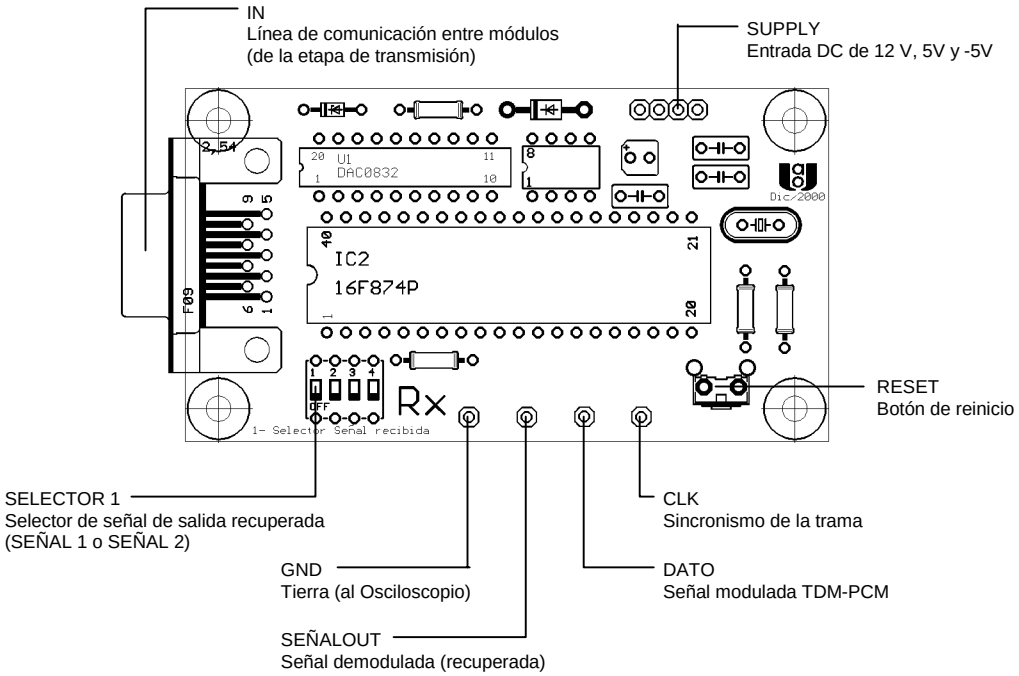


FIGURA 2 *Etapa de recepción del módulo PCM*

2.1 Conectores

SUPPLY:

Esta es la entrada a la cual se debe conectar la fuente de poder que genera tres voltajes diferentes (12, 5 y -5 voltios), estos se encargan de alimentar la tarjeta.

IN:

Es la línea de comunicación entre las etapas de transmisión y recepción del módulo, esta línea viene de la etapa de transmisión.

2.2 Controles

RESET:

Este control es un micropulsador, el cual permite reiniciar el programa que esta corriendo en el microcontrolador de la etapa de recepción, esto en caso que sea necesario hacerlo.

2.3 Selectores

SELECTOR 1:

Este selector es un switch de dos posiciones, el cual permite seleccionar la señal análoga que va a ser recuperada. Solo una de las dos señales análogas (SEÑAL 1 o SEÑAL 2) se recupera y esta se puede observar en el terminal señalado con el nombre SEÑALOUT.

2.4 Puntos de prueba

GND:

Tierra, terminal común de la etapa de recepción, este se debe conectar a la tierra del Osciloscopio.

SEÑALOUT:

En este punto se obtiene la señal demodulada (recuperada) de una de las dos señales análogas (SEÑAL 1 o SEÑAL 2) generadas en la etapa de transmisión.

DATO:

Este punto permite observar la señal modulada, es decir, la señal PCM que corresponde a la muestra de la señal análoga, en este caso se observaran dos señales PCM multiplexadas (trama TDM-PCM) ya que son dos señales análogas.

CLK:

En este terminal se puede observar la señal de sincronización que acompaña la trama TDM-PCM, ya que se utiliza comunicación sincrónica.

3. FUNCIONAMIENTO DEL MODULO

3.1 Conexión

1. Conecte la fuente de poder, la cual genera tres voltajes diferentes (12, 5 y -5 voltios) que se utilizan como alimentación, al sitio indicado con el nombre SUPPLY en cada una de las tarjetas de las etapas.

2. Conecte la línea de comunicación entre la etapa de transmisión y la etapa de recepción. Esta línea se conecta entre el sitio señalado con el nombre OUT en la etapa de transmisión y el indicado con el nombre IN en la etapa de recepción.

3. Conecte el terminal señalado con el nombre GND al terminal de tierra del Osciloscopio, de esta manera se podrán observar las señales generadas por las etapas de los módulos.

4. Envié la punta de prueba del Osciloscopio a uno de los puntos donde se obtienen las señales a observar en las etapas.

Si después de conectar y activar las etapas de transmisión y recepción del módulo, no se observan las señales que se esperan en los puntos de prueba, reinicie los programas que corren en los microcontroladores de cada etapa utilizando el botón de reinicio (RESET).

3.2 Entrada análoga

La entrada análoga es generada en la etapa de transmisión con ayuda de un generador de funciones XR-2206, en el caso del sistema PCM se utilizan dos generadores para producir dos señales, estas señales se pueden modificar en frecuencia y amplitud por medio de un juego de potenciómetros (Trimmers). La señal de cada generador puede tener dos formas de onda diferentes, sinusoidal o triangular, la forma de onda que adoptara la señal se selecciona por medio de un switch de dos posiciones.

Las señales análogas (SEÑAL 1 y SEÑAL 2) que producen los generadores se pueden obtener de los pines dos y tres respectivamente del microcontrolador PIC16F874.

GUIA DE USO DEL MODULO DELTA

A continuación se describen cada una de las partes de las etapas de transmisión y recepción que utiliza el sistema DELTA, así como su forma de conexión.

1. LOCALIZACION DE PARTES EN LA ETAPA DE TRANSMISION

La tarjeta que genera la señal análoga de entrada y realiza el proceso de modulación de la misma esta compuesta como se aprecia en la Figura 1.

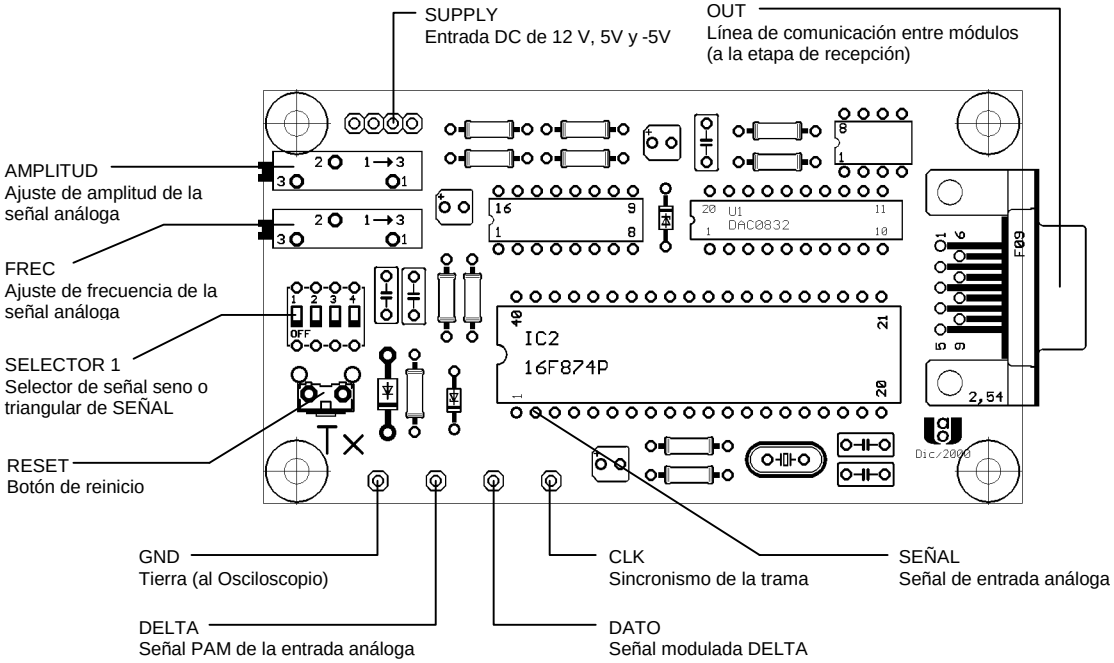


FIGURA 1 *Etapa de transmisión del módulo DELTA*

1.1 Conectores

SUPPLY:

Esta es la entrada a la cual se debe conectar la fuente de poder que genera tres voltajes diferentes (12, 5 y -5 voltios), estos se encargan de alimentar la tarjeta.

OUT:

Es la línea de comunicación entre las etapas de transmisión y recepción del módulo, esta línea va a la etapa de recepción.

1.2 Controles

RESET:

Este control es un micropulsador, el cual permite reiniciar el programa que esta corriendo en el microcontrolador de la etapa de transmisión, esto en caso que sea necesario hacerlo.

AMPLITUD:

Este control es un potenciómetro (Trimmer), el cual permite ajustar la amplitud de la señal análoga. La máxima amplitud es de cinco voltios.

FREC:

Este control es un potenciómetro (Trimmer) que permite realizar el ajuste de la frecuencia de la señal análoga, esta frecuencia se puede variar entre el rango de 15 y 30 Hz.

1.3 Selectores

SELECTOR 1:

Este selector es un switch de dos posiciones, el cual permite elegir la forma de onda de la señal análoga, es decir, la señal será una onda seno o una triangular.

1.4 Puntos de prueba

GND:

Tierra, terminal común de la etapa de transmisión, este se debe conectar a la tierra del Osciloscopio.

DELTA:

En este punto se obtiene la señal PAM correspondiente a la señal análoga que se usa como entrada de la etapa de transmisión.

DATO:

Este punto permite observar la señal modulada, es decir, la señal DELTA que corresponde a la muestra de la señal análoga. Esta señal es la que se envía a la etapa de recepción a través de la línea de comunicación.

CLK:

En este terminal se puede observar la señal de sincronización que acompaña la trama DELTA, ya que se utiliza comunicación sincrónica.

La señal análoga se obtiene en el terminal dos del microcontrolador PIC16F874.

2. LOCALIZACION DE PARTES EN LA ETAPA DE RECEPCION

La tarjeta que recibe la señal de la etapa de transmisión y realiza el proceso de demodulación esta compuesta como se aprecia en la Figura 2.

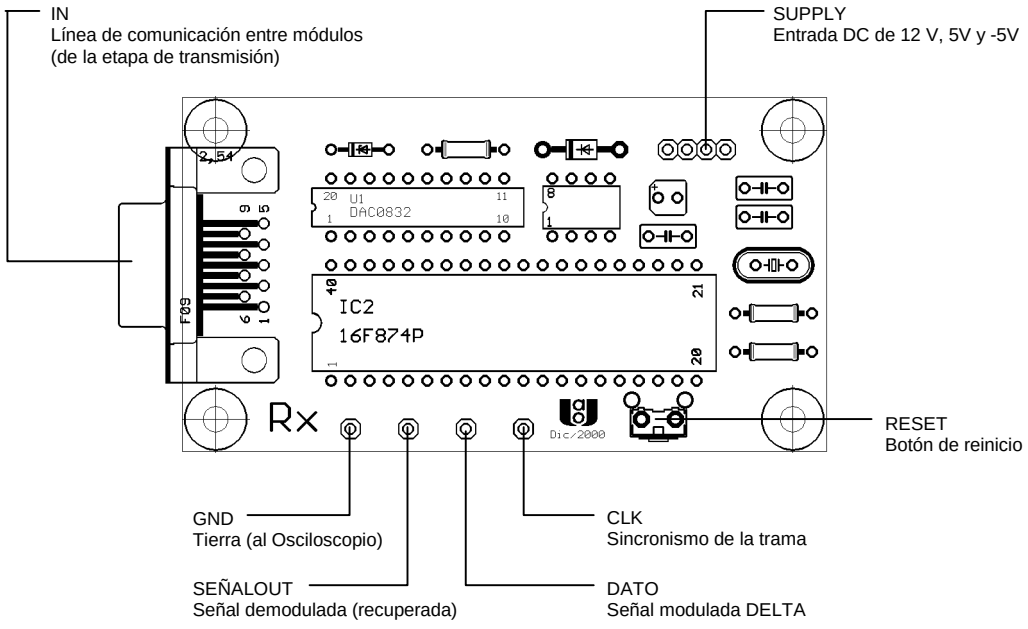


FIGURA 2 *Etapa de recepción del módulo DELTA*

2.1 Conectores

SUPPLY:
Esta es la entrada a la cual se debe conectar la fuente de poder que genera tres voltajes diferentes (12, 5 y -5 voltios), estos se encargan de alimentar la tarjeta.

IN:
Es la línea de comunicación entre las etapas de transmisión y recepción del módulo, esta línea viene de la etapa de transmisión.

2.2 Controles

RESET:
Este control es un micropulsador, el cual permite reiniciar el programa que esta corriendo en el microcontrolador de la etapa de recepción, esto en caso que sea necesario hacerlo.

2.3 Puntos de prueba

GND:
Tierra, terminal común de la etapa de recepción, este se debe conectar a la tierra del Osciloscopio.

SEÑALOUT:
En este punto se obtiene la señal demodulada (recuperada) de la señal análoga generada en la etapa de transmisión.

DATO:
Este punto permite observar la señal modulada, es decir, la señal DELTA que corresponde a la muestra de la señal análoga.

CLK:
En este terminal se puede observar la señal de sincronización que acompaña la trama DELTA, ya que se utiliza comunicación sincrónica.

3. FUNCIONAMIENTO DEL MODULO

3.1 Conexión

1. Conecte la fuente de poder, la cual genera tres voltajes diferentes (12, 5 y -5 voltios) que se utilizan como alimentación, al sitio indicado con el nombre SUPPLY en cada una de las tarjetas de las etapas.

2. Conecte la línea de comunicación entre la etapa de transmisión y la etapa de recepción. Esta línea se conecta entre el sitio señalado con el nombre OUT en la etapa de transmisión y el indicado con el nombre IN en la etapa de recepción.

3. Conecte el terminal señalado con el nombre GND al terminal de tierra del Osciloscopio, de esta manera se podrán observar las señales generadas por las etapas de los módulos.

4. Envié la punta de prueba del Osciloscopio a uno de los puntos donde se obtienen las señales a observar en las etapas.

Si después de conectar y activar las etapas de transmisión y recepción del módulo, no se observan las señales que se esperan en los puntos de prueba, reinicie los programas que corren en los microcontroladores de cada etapa utilizando el botón de reinicio (RESET).

3.2 Entrada análoga

La entrada análoga es generada en la etapa de transmisión con ayuda de un generador de funciones XR-2206, esta señal se puede modificar en frecuencia y amplitud por medio de un par de potenciómetros (Trimmers). La señal del generador puede tener dos formas de onda diferentes, sinusoidal o triangular, la forma de onda que adoptara la señal se selecciona por medio de un switch de dos posiciones. La señal análoga que produce el generador se puede obtener del pin dos del microcontrolador PIC16F874.